http://www.romanpichler.com/blog/user-stories/writing-good-user-stories/

10 Tips for Writing Good User Stories

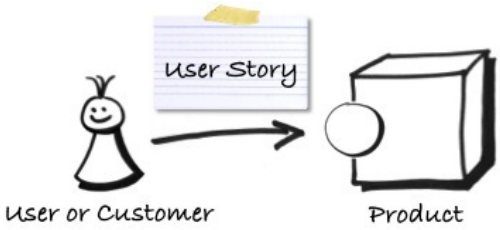
Posted on Thursday 7th October 2010

Summary

User stories are probably the most popular agile technique to capture product functionality: Working with user stories is easy. But writing good stories can be hard. The following ten tips help you create good stories.

1. Focus on the user

As its name suggests, a user story tells a story about a customer or user employing the product. Write therefore stories from the user's perspective and show how a user or customer uses some product functionality.

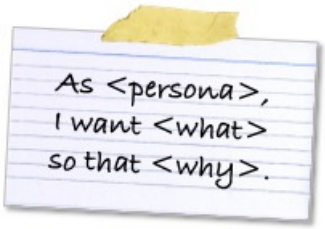


2. Write stories collaboratively

A user story is not a specification, but an communication and collaboration tool. Stories should not be handed off to the development team but be complemented by a conversation: Product owner and team should talk about stories, or even better, write them together: Invite the team to create and detail the stories together. This leverages the creativity and the knowledge of the team and usually results in better user stories.

3. Keep your stories simple and concise

Write your stories so that they are easy to understand. Keep them simple and concise. Avoid confusing and ambiguous terms, and use active voice. Focus on what's important, and leave out non-essential information. The following template puts the user or customer into the story and makes its benefit explicit. But use the template only if it is helpful. Try out different ways to write your stories to understand what works best for you.

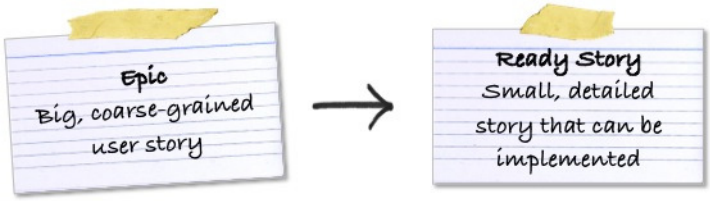


4. Start with epics

Epics are big, coarse-grained user stories. Starting with epics allows you to sketch the product functionality without committing to the details. This is particularly helpful for new products and new features, as it buys you time to learn more about the users and how to best meet their needs. It also reduces the time and effort required to integrate new insights: If you have lots of detailed stories, then it's often tricky to relate the feedback you receive to the right stories. I like to use the personas with their goals to discover the right epics.

5. Decompose your stories until they are ready

Break your epics into smaller, detailed stories until they are ready: clear, feasible, and testable. Everyone should have a shared understanding of the story's meaning; the story should not too big, and there has to be an effective way to determine if the story is done.



6. Add acceptance criteria

As you decompose epics into smaller stories, remember to add acceptance criteria. Acceptance criteria complement the story's narrative: They allow you to describe the conditions that have to be fulfilled so that the story is done. The criteria enrich the story and make it more precise and testable, and they ensures that the story can be demoed or released to the users and the other stakeholders.

7. Group user stories into themes

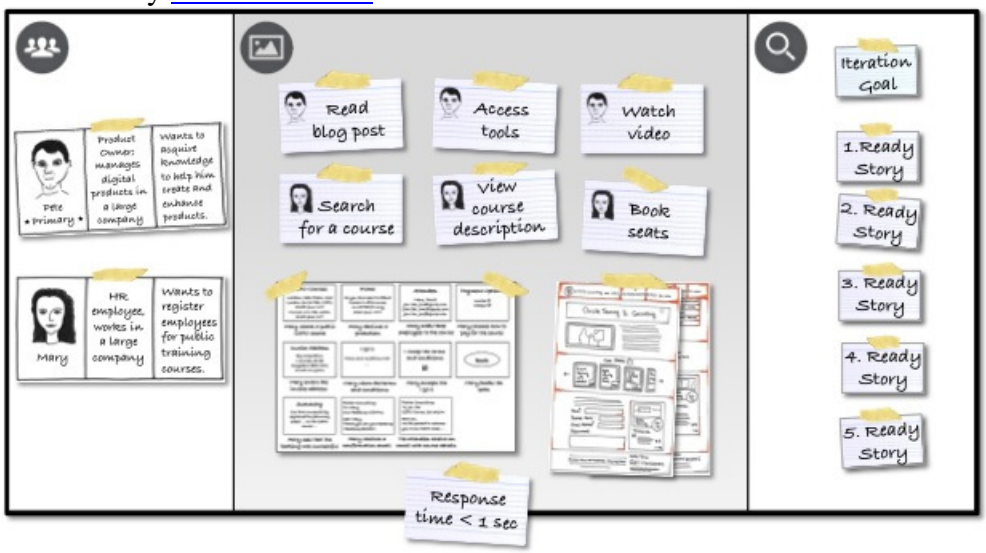
Using themes helps you organise your stories. Each theme is a group of related stories. Sample themes for a mobile phone are email, calendar, voice communication, and organizer, for instance. Themes make it easier to check for completeness and consistency, they structure your product canvas or product backlog, and they facilitate prioritisation.

8. Use paper cards

Paper cards are not only cheap and easy to use. They facilitate collaboration, as everyone can take a card and jot down an idea. Cards can also be easily grouped on the table or wall to check for consistency and completeness. Even if your stories are stored electronically, it is worthwhile to use paper cards when you write new stories.

9. Keep your stories visible and accessible

Stories want to communication information. Don't hide them on a network drive, the corporate intranet jungle, or a licensed tool. Make them visible instead, for instance, by putting them up on the wall. A great tool to discover, visualise, and manage your stories is my Product Canvas.



10. Don't solely rely on user stories

Creating a great user experience (UX) requires more than writing user stories. Also consider the user journeys and interactions, the visual design, and the nonfunctional properties of your product. This results in a holistic description that makes it easier to identify risks and assumptions, and it increases the chances of creating a product with the right user experience.

http://www.romanpichler.com/blog/user-stories/5-common-user-story-mistakes/

5 Common User Story Mistakes

Posted on Wednesday 12th June 2013

Summary

This post discusses five common user story mistakes and helps you overcome them.

User stories are a simple, yet effective way to communicate how a user or customer employs a product. But writing user stories that help a team build great software can be challenging. The following list summarises five common user story mistakes to help you improve your stories.

Story Mania

Some product owners and teams are so fond of user stories that everything is expressed as a story. This either results in some rather odd stories – stories that capture the user interface design, complex user interactions, and technical requirements; or these aspects are simply overlooked.

Like any technique, user story writing has its strengths and limitations. I find stories particularly well suited to capture product functionality, and when applied properly, nonfunctional requirements. But user interface design and complex user interactions are better described by other techniques including design sketches, mock-ups, scenarios, and storyboards. Complement your user stories therefore with other techniques, and don't feel obliged to only use stories.

User Incognito

A user story tells a story about a user interacting with the product. Some stories, however, omit the beneficiary altogether, or they talk about a mysterious user as in "As the user, I want to …" But if it's not clear who the user is, why should the story be developed? How can we be confident that the story will benefit someone?

Make sure that all your stories have a clear user or customer. As I like to work with personas, I use personas in my stories (instead of user roles). This connects each story to the right persona, and it allows me to understand if and to which extend the story addresses the persona's need or goal. To achieve this, I use the following template: As <persona>, I want <something> so that <benefit>.

Disastrous Details

The devil is in the details: Some stories are too big and vague for the team to understand and implement. Others contain too much detail, or even prescribe a solution. Getting the details right seems to be a battle the product owner can only loose.

My recommendation is: Start with big, coarse-grained stories called epics particularly for all new features. Epics allow you to capture an idea without committing to the details. Then break an epic into more detailed user stories. The new user stories replace the epic, and they provide more information about the product's functionality. Pay particular attention to the user stories that are pulled into a sprint. These stories have to be ready: clear, feasible, and testable. (You can learn more about decomposing epics into ready stories in my post "Epics and Ready Stories".)

Progressively refining your stories keeps your Product Canvas or backlog concise, it makes it easier to integrate new insights, and it reduces the effort required to stock your initial canvas or backlog. This is particularly valuable for new products and new features. Make sure you do not prescribe a solution in your stories. Rather focus on the "what", nor the "how". The latter is best captured in an architecture diagram.

Story Handoff

Some product owners diligently write user stories, and give them to the development team in the sprint planning meeting. This handoff is usually suboptimal, as it wastes the team's ideas and knowledge. Stories can hence be inappropriate, difficult to understand, unfeasible and not testable.

User stories, however, are not meant to be standalone documents. They should be complemented by a conversation between the product owner and the team, or even better: written collaboratively. A story wants to capture the essentials, and not specify every detail: The latter would be difficult for new features and too slow and too expensive in an agile / lean context.

User story writing should be a team effort, where product owner and development team create the stories together. Allocate time for a collaborative Product Canvas workshop or backlog grooming meeting, particularly when you develop a new product or new features. Trust me: Better stories and a better product will be your reward.

Criteria Crisis

Acceptance criteria are maybe the most misunderstood part of users stories. I have seen detailed stories with no acceptance criteria, criteria that restate the narrative, and criteria that hide new stories, or even contain entire workflows. The last two mistakes are exemplified by the following story (the narrative is on the card on the left, the "acceptance criteria" on the right):

The idea behind acceptance criteria is simple: They allow you to describe the conditions that have to be fulfilled so that the story is done, that it can be exposed to the users and the other stakeholders. This ensures that you gather feedback and/or release features, and it helps the team plan and track their work: The criteria enrich the story and make it more precise and testable. (The criteria all stories have to fulfil such as "online help is available" are not stated in the acceptance criteria but in the Definition of Done.)

As a rule of thumb, I like to work with three to five criteria per story, and I am not worried if my epics don't have acceptance criteria to start with. Ready stories, however, must provide meaningful criteria. You can find sample acceptance criteria in my posts "Epics and Ready Stories" and "Nonfunctional Requirements".

http://www.romanpichler.com/blog/user-stories/writing-good-user-stories/

10 Tips for Writing Good User Stories

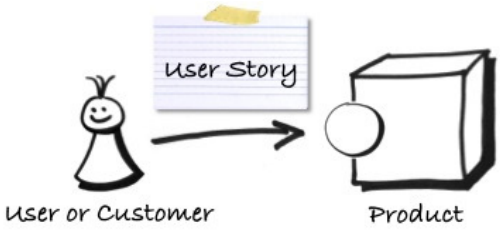
Posted on Thursday 7th October 2010

Summary

User stories are probably the most popular agile technique to capture product functionality: Working with user stories is easy. But writing good stories can be hard. The following ten tips help you create good stories.

1. Focus on the user

As its name suggests, a user story tells a story about a customer or user employing the product. Write therefore stories from the user's perspective and show how a user or customer uses some product functionality.

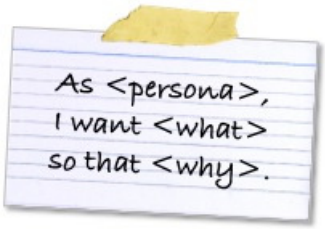


2. Write stories collaboratively

A user story is not a specification, but an communication and collaboration tool. Stories should not be handed off to the development team but be complemented by a conversation: Product owner and team should talk about stories, or even better, write them together: Invite the team to create and detail the stories together. This leverages the creativity and the knowledge of the team and usually results in better user stories.

3. Keep your stories simple and concise

Write your stories so that they are easy to understand. Keep them simple and concise. Avoid confusing and ambiguous terms, and use active voice. Focus on what's important, and leave out non-essential information. The following template puts the user or customer into the story and makes its benefit explicit. But use the template only if it is helpful. Try out different ways to write your stories to understand what works best for you.

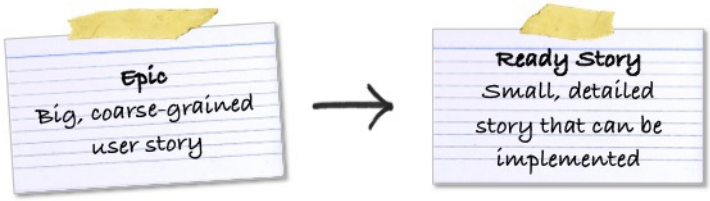


4. Start with epics

Epics are big, coarse-grained user stories. Starting with epics allows you to sketch the product functionality without committing to the details. This is particularly helpful for new products and new features, as it buys you time to learn more about the users and how to best meet their needs. It also reduces the time and effort required to integrate new insights: If you have lots of detailed stories, then it's often tricky to relate the feedback you receive to the right stories. I like to use the personas with their goals to discover the right epics.

5. Decompose your stories until they are ready

Break your epics into smaller, detailed stories until they are ready: clear, feasible, and testable. Everyone should have a shared understanding of the story's meaning; the story should not too big, and there has to be an effective way to determine if the story is done.



6. Add acceptance criteria

As you decompose epics into smaller stories, remember to add acceptance criteria. Acceptance criteria complement the story's narrative: They allow you to describe the conditions that have to be fulfilled so that the story is done. The criteria enrich the story and make it more precise and testable, and they ensures that the story can be demoed or released to the users and the other stakeholders.

7. Group user stories into themes

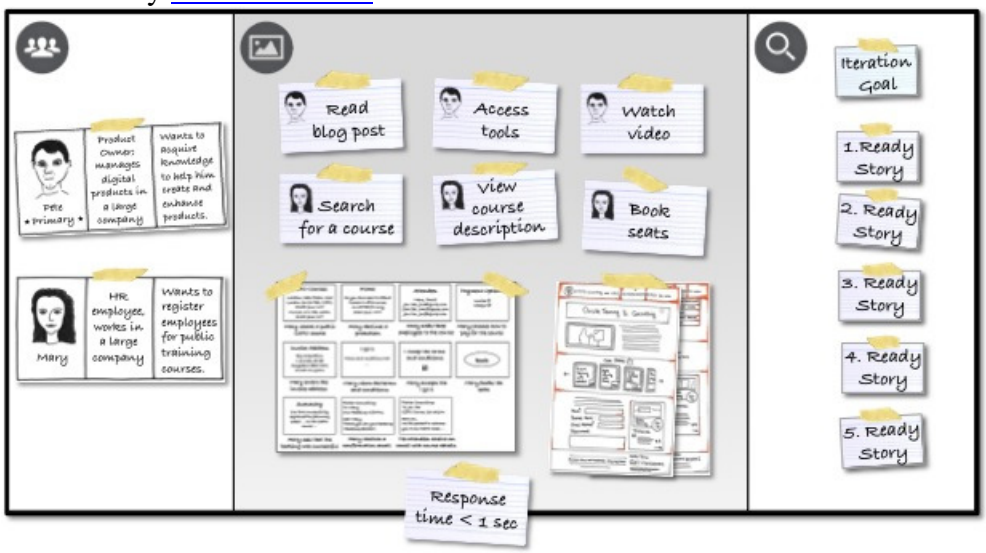
Using themes helps you organise your stories. Each theme is a group of related stories. Sample themes for a mobile phone are email, calendar, voice communication, and organizer, for instance. Themes make it easier to check for completeness and consistency, they structure your product canvas or product backlog, and they facilitate prioritisation.

8. Use paper cards

Paper cards are not only cheap and easy to use. They facilitate collaboration, as everyone can take a card and jot down an idea. Cards can also be easily grouped on the table or wall to check for consistency and completeness. Even if your stories are stored electronically, it is worthwhile to use paper cards when you write new stories.

9. Keep your stories visible and accessible

Stories want to communication information. Don't hide them on a network drive, the corporate intranet jungle, or a licensed tool. Make them visible instead, for instance, by putting them up on the wall. A great tool to discover, visualise, and manage your stories is my Product Canvas.



10. Don't solely rely on user stories

Creating a great user experience (UX) requires more than writing user stories. Also consider the user journeys and interactions, the visual design, and the nonfunctional properties of your product. This results in a holistic description that makes it easier to identify risks and assumptions, and it increases the chances of creating a product with the right user experience.

http://www.romanpichler.com/blog/user-stories/5-common-user-story-mistakes/

5 Common User Story Mistakes

Posted on Wednesday 12th June 2013

Summary

This post discusses five common user story mistakes and helps you overcome them.

User stories are a simple, yet effective way to communicate how a user or customer employs a product. But writing user stories that help a team build great software can be challenging. The following list summarises five common user story mistakes to help you improve your stories.

Story Mania

Some product owners and teams are so fond of user stories that everything is expressed as a story. This either results in some rather odd stories – stories that capture the user interface design, complex user interactions, and technical requirements; or these aspects are simply overlooked.

Like any technique, user story writing has its strengths and limitations. I find stories particularly well suited to capture product functionality, and when applied properly, nonfunctional requirements. But user interface design and complex user interactions are better described by other techniques including design sketches, mock-ups, scenarios, and storyboards. Complement your user stories therefore with other techniques, and don't feel obliged to only use stories.

User Incognito

A user story tells a story about a user interacting with the product. Some stories, however, omit the beneficiary altogether, or they talk about a mysterious user as in "As the user, I want to …" But if it's not clear who the user is, why should the story be developed? How can we be confident that the story will benefit someone?

Make sure that all your stories have a clear user or customer. As I like to work with personas, I use personas in my stories (instead of user roles). This connects each story to the right persona, and it allows me to understand if and to which extend the story addresses the persona's need or goal. To achieve this, I use the following template: As <persona>, I want <something> so that <benefit>.

Disastrous Details

The devil is in the details: Some stories are too big and vague for the team to understand and implement. Others contain too much detail, or even prescribe a solution. Getting the details right seems to be a battle the product owner can only loose.

My recommendation is: Start with big, coarse-grained stories called epics particularly for all new features. Epics allow you to capture an idea without committing to the details. Then break an epic into more detailed user stories. The new user stories replace the epic, and they provide more information about the product's functionality. Pay particular attention to the user stories that are pulled into a sprint. These stories have to be ready: clear, feasible, and testable. (You can learn more about decomposing epics into ready stories in my post "Epics and Ready Stories".)

Progressively refining your stories keeps your Product Canvas or backlog concise, it makes it easier to integrate new insights, and it reduces the effort required to stock your initial canvas or backlog. This is particularly valuable for new products and new features. Make sure you do not prescribe a solution in your stories. Rather focus on the "what", nor the "how". The latter is best captured in an architecture diagram.

Story Handoff

Some product owners diligently write user stories, and give them to the development team in the sprint planning meeting. This handoff is usually suboptimal, as it wastes the team's ideas and knowledge. Stories can hence be inappropriate, difficult to understand, unfeasible and not testable.

User stories, however, are not meant to be standalone documents. They should be complemented by a conversation between the product owner and the team, or even better: written collaboratively. A story wants to capture the essentials, and not specify every detail: The latter would be difficult for new features and too slow and too expensive in an agile / lean context.

User story writing should be a team effort, where product owner and development team create the stories together. Allocate time for a collaborative Product Canvas workshop or backlog grooming meeting, particularly when you develop a new product or new features. Trust me: Better stories and a better product will be your reward.

Criteria Crisis

Acceptance criteria are maybe the most misunderstood part of users stories. I have seen detailed stories with no acceptance criteria, criteria that restate the narrative, and criteria that hide new stories, or even contain entire workflows. The last two mistakes are exemplified by the following story (the narrative is on the card on the left, the "acceptance criteria" on the right):

The idea behind acceptance criteria is simple: They allow you to describe the conditions that have to be fulfilled so that the story is done, that it can be exposed to the users and the other stakeholders. This ensures that you gather feedback and/or release features, and it helps the team plan and track their work: The criteria enrich the story and make it more precise and testable. (The criteria all stories have to fulfil such as "online help is available" are not stated in the acceptance criteria but in the Definition of Done.)

As a rule of thumb, I like to work with three to five criteria per story, and I am not worried if my epics don't have acceptance criteria to start with. Ready stories, however, must provide meaningful criteria. You can find sample acceptance criteria in my posts "Epics and Ready Stories" and "Nonfunctional Requirements".

http://www.romanpichler.com/blog/user-stories/writing-good-user-stories/

10 Tips for Writing Good User Stories

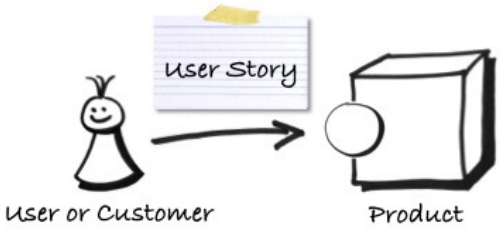
Posted on Thursday 7th October 2010

Summary

User stories are probably the most popular agile technique to capture product functionality: Working with user stories is easy. But writing good stories can be hard. The following ten tips help you create good stories.

1. Focus on the user

As its name suggests, a user story tells a story about a customer or user employing the product. Write therefore stories from the user's perspective and show how a user or customer uses some product functionality.

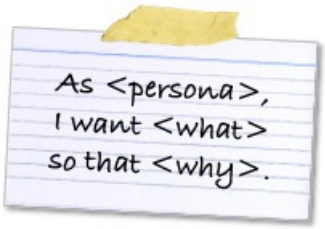


2. Write stories collaboratively

A user story is not a specification, but an communication and collaboration tool. Stories should not be handed off to the development team but be complemented by a conversation: Product owner and team should talk about stories, or even better, write them together: Invite the team to create and detail the stories together. This leverages the creativity and the knowledge of the team and usually results in better user stories.

3. Keep your stories simple and concise

Write your stories so that they are easy to understand. Keep them simple and concise. Avoid confusing and ambiguous terms, and use active voice. Focus on what's important, and leave out non-essential information. The following template puts the user or customer into the story and makes its benefit explicit. But use the template only if it is helpful. Try out different ways to write your stories to understand what works best for you.

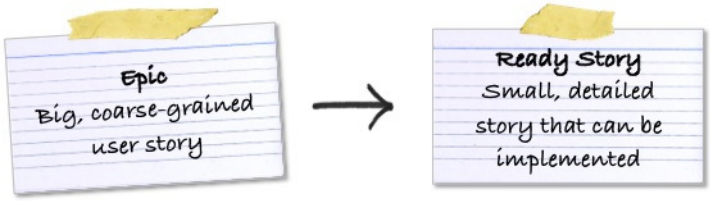


4. Start with epics

Epics are big, coarse-grained user stories. Starting with epics allows you to sketch the product functionality without committing to the details. This is particularly helpful for new products and new features, as it buys you time to learn more about the users and how to best meet their needs. It also reduces the time and effort required to integrate new insights: If you have lots of detailed stories, then it's often tricky to relate the feedback you receive to the right stories. I like to use the personas with their goals to discover the right epics.

5. Decompose your stories until they are ready

Break your epics into smaller, detailed stories until they are ready: clear, feasible, and testable. Everyone should have a shared understanding of the story's meaning; the story should not too big, and there has to be an effective way to determine if the story is done.



6. Add acceptance criteria

As you decompose epics into smaller stories, remember to add acceptance criteria. Acceptance criteria complement the story's narrative: They allow you to describe the conditions that have to be fulfilled so that the story is done. The criteria enrich the story and make it more precise and testable, and they ensures that the story can be demoed or released to the users and the other stakeholders.

7. Group user stories into themes

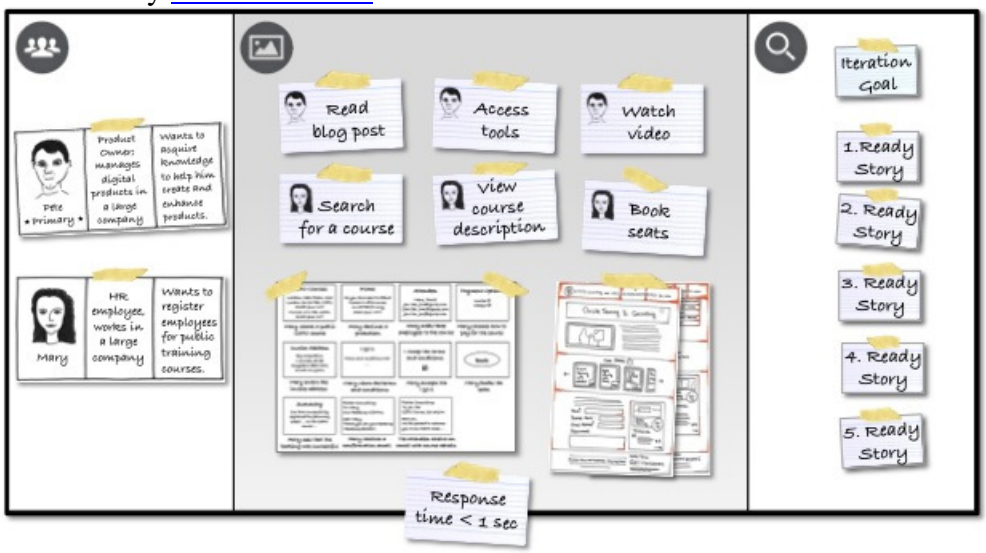
Using themes helps you organise your stories. Each theme is a group of related stories. Sample themes for a mobile phone are email, calendar, voice communication, and organizer, for instance. Themes make it easier to check for completeness and consistency, they structure your product canvas or product backlog, and they facilitate prioritisation.

8. Use paper cards

Paper cards are not only cheap and easy to use. They facilitate collaboration, as everyone can take a card and jot down an idea. Cards can also be easily grouped on the table or wall to check for consistency and completeness. Even if your stories are stored electronically, it is worthwhile to use paper cards when you write new stories.

9. Keep your stories visible and accessible

Stories want to communication information. Don't hide them on a network drive, the corporate intranet jungle, or a licensed tool. Make them visible instead, for instance, by putting them up on the wall. A great tool to discover, visualise, and manage your stories is my Product Canvas.



10. Don't solely rely on user stories

Creating a great user experience (UX) requires more than writing user stories. Also consider the user journeys and interactions, the visual design, and the nonfunctional properties of your product. This results in a holistic description that makes it easier to identify risks and assumptions, and it increases the chances of creating a product with the right user experience.

http://www.romanpichler.com/blog/user-stories/5-common-user-story-mistakes/

5 Common User Story Mistakes

Posted on Wednesday 12th June 2013

Summary

This post discusses five common user story mistakes and helps you overcome them.

User stories are a simple, yet effective way to communicate how a user or customer employs a product. But writing user stories that help a team build great software can be challenging. The following list summarises five common user story mistakes to help you improve your stories.

Story Mania

Some product owners and teams are so fond of user stories that everything is expressed as a story. This either results in some rather odd stories – stories that capture the user interface design, complex user interactions, and technical requirements; or these aspects are simply overlooked.

Like any technique, user story writing has its strengths and limitations. I find stories particularly well suited to capture product functionality, and when applied properly, nonfunctional requirements. But user interface design and complex user interactions are better described by other techniques including design sketches, mock-ups, scenarios, and storyboards. Complement your user stories therefore with other techniques, and don't feel obliged to only use stories.

User Incognito

A user story tells a story about a user interacting with the product. Some stories, however, omit the beneficiary altogether, or they talk about a mysterious user as in "As the user, I want to …" But if it's not clear who the user is, why should the story be developed? How can we be confident that the story will benefit someone?

Make sure that all your stories have a clear user or customer. As I like to work with personas, I use personas in my stories (instead of user roles). This connects each story to the right persona, and it allows me to understand if and to which extend the story addresses the persona's need or goal. To achieve this, I use the following template: As <persona>, I want <something> so that <benefit>.

Disastrous Details

The devil is in the details: Some stories are too big and vague for the team to understand and implement. Others contain too much detail, or even prescribe a solution. Getting the details right seems to be a battle the product owner can only loose.

My recommendation is: Start with big, coarse-grained stories called epics particularly for all new features. Epics allow you to capture an idea without committing to the details. Then break an epic into more detailed user stories. The new user stories replace the epic, and they provide more information about the product's functionality. Pay particular attention to the user stories that are pulled into a sprint. These stories have to be ready: clear, feasible, and testable. (You can learn more about decomposing epics into ready stories in my post "Epics and Ready Stories".)

Progressively refining your stories keeps your Product Canvas or backlog concise, it makes it easier to integrate new insights, and it reduces the effort required to stock your initial canvas or backlog. This is particularly valuable for new products and new features. Make sure you do not prescribe a solution in your stories. Rather focus on the "what", nor the "how". The latter is best captured in an architecture diagram.

Story Handoff

Some product owners diligently write user stories, and give them to the development team in the sprint planning meeting. This handoff is usually suboptimal, as it wastes the team's ideas and knowledge. Stories can hence be inappropriate, difficult to understand, unfeasible and not testable.

User stories, however, are not meant to be standalone documents. They should be complemented by a conversation between the product owner and the team, or even better: written collaboratively. A story wants to capture the essentials, and not specify every detail: The latter would be difficult for new features and too slow and too expensive in an agile / lean context.

User story writing should be a team effort, where product owner and development team create the stories together. Allocate time for a collaborative Product Canvas workshop or backlog grooming meeting, particularly when you develop a new product or new features. Trust me: Better stories and a better product will be your reward.

Criteria Crisis

Acceptance criteria are maybe the most misunderstood part of users stories. I have seen detailed stories with no acceptance criteria, criteria that restate the narrative, and criteria that hide new stories, or even contain entire workflows. The last two mistakes are exemplified by the following story (the narrative is on the card on the left, the "acceptance criteria" on the right):

The idea behind acceptance criteria is simple: They allow you to describe the conditions that have to be fulfilled so that the story is done, that it can be exposed to the users and the other stakeholders. This ensures that you gather feedback and/or release features, and it helps the team plan and track their work: The criteria enrich the story and make it more precise and testable. (The criteria all stories have to fulfil such as "online help is available" are not stated in the acceptance criteria but in the Definition of Done.)

As a rule of thumb, I like to work with three to five criteria per story, and I am not worried if my epics don't have acceptance criteria to start with. Ready stories, however, must provide meaningful criteria. You can find sample acceptance criteria in my posts "Epics and Ready Stories" and "Nonfunctional Requirements".

http://www.romanpichler.com/blog/user-stories/writing-good-user-stories/

10 Tips for Writing Good User Stories

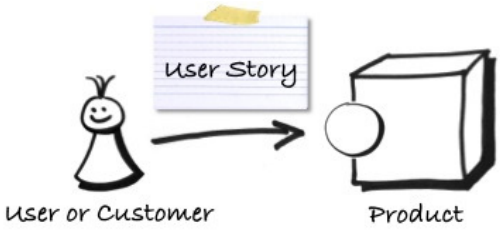
Posted on Thursday 7th October 2010

Summary

User stories are probably the most popular agile technique to capture product functionality: Working with user stories is easy. But writing good stories can be hard. The following ten tips help you create good stories.

1. Focus on the user

As its name suggests, a user story tells a story about a customer or user employing the product. Write therefore stories from the user's perspective and show how a user or customer uses some product functionality.

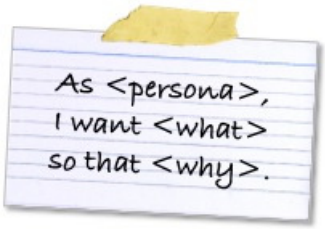


2. Write stories collaboratively

A user story is not a specification, but an communication and collaboration tool. Stories should not be handed off to the development team but be complemented by a conversation: Product owner and team should talk about stories, or even better, write them together: Invite the team to create and detail the stories together. This leverages the creativity and the knowledge of the team and usually results in better user stories.

3. Keep your stories simple and concise

Write your stories so that they are easy to understand. Keep them simple and concise. Avoid confusing and ambiguous terms, and use active voice. Focus on what's important, and leave out non-essential information. The following template puts the user or customer into the story and makes its benefit explicit. But use the template only if it is helpful. Try out different ways to write your stories to understand what works best for you.

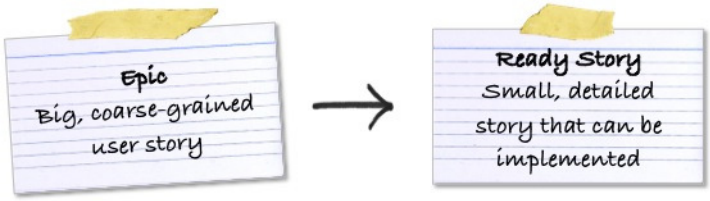


4. Start with epics

Epics are big, coarse-grained user stories. Starting with epics allows you to sketch the product functionality without committing to the details. This is particularly helpful for new products and new features, as it buys you time to learn more about the users and how to best meet their needs. It also reduces the time and effort required to integrate new insights: If you have lots of detailed stories, then it's often tricky to relate the feedback you receive to the right stories. I like to use the personas with their goals to discover the right epics.

5. Decompose your stories until they are ready

Break your epics into smaller, detailed stories until they are ready: clear, feasible, and testable. Everyone should have a shared understanding of the story's meaning; the story should not too big, and there has to be an effective way to determine if the story is done.



6. Add acceptance criteria

As you decompose epics into smaller stories, remember to add acceptance criteria. Acceptance criteria complement the story's narrative: They allow you to describe the conditions that have to be fulfilled so that the story is done. The criteria enrich the story and make it more precise and testable, and they ensures that the story can be demoed or released to the users and the other stakeholders.

7. Group user stories into themes

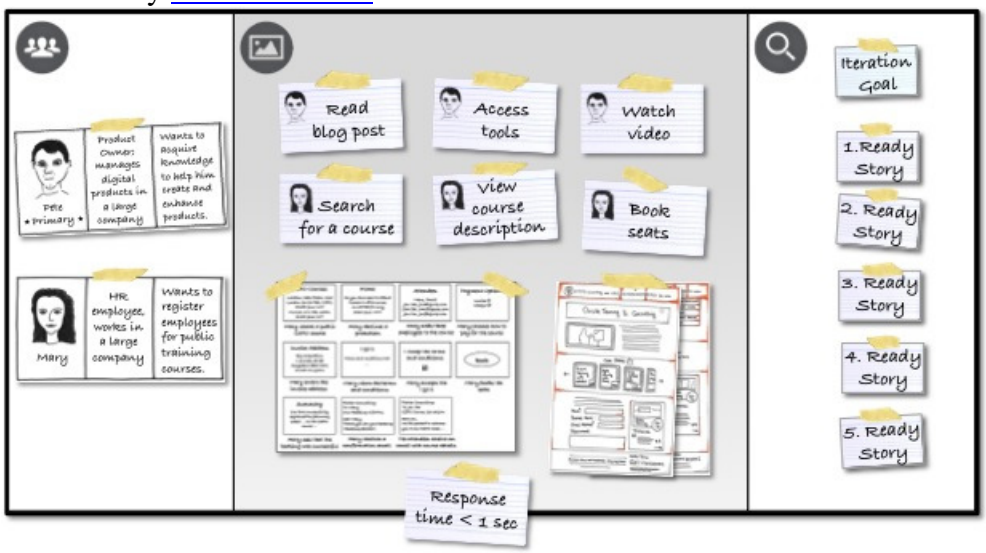
Using themes helps you organise your stories. Each theme is a group of related stories. Sample themes for a mobile phone are email, calendar, voice communication, and organizer, for instance. Themes make it easier to check for completeness and consistency, they structure your product canvas or product backlog, and they facilitate prioritisation.

8. Use paper cards

Paper cards are not only cheap and easy to use. They facilitate collaboration, as everyone can take a card and jot down an idea. Cards can also be easily grouped on the table or wall to check for consistency and completeness. Even if your stories are stored electronically, it is worthwhile to use paper cards when you write new stories.

9. Keep your stories visible and accessible

Stories want to communication information. Don't hide them on a network drive, the corporate intranet jungle, or a licensed tool. Make them visible instead, for instance, by putting them up on the wall. A great tool to discover, visualise, and manage your stories is my Product Canvas.



10. Don't solely rely on user stories

Creating a great user experience (UX) requires more than writing user stories. Also consider the user journeys and interactions, the visual design, and the nonfunctional properties of your product. This results in a holistic description that makes it easier to identify risks and assumptions, and it increases the chances of creating a product with the right user experience.

http://www.romanpichler.com/blog/user-stories/5-common-user-story-mistakes/

5 Common User Story Mistakes

Posted on Wednesday 12th June 2013

Summary

This post discusses five common user story mistakes and helps you overcome them.

User stories are a simple, yet effective way to communicate how a user or customer employs a product. But writing user stories that help a team build great software can be challenging. The following list summarises five common user story mistakes to help you improve your stories.

Story Mania

Some product owners and teams are so fond of user stories that everything is expressed as a story. This either results in some rather odd stories – stories that capture the user interface design, complex user interactions, and technical requirements; or these aspects are simply overlooked.

Like any technique, user story writing has its strengths and limitations. I find stories particularly well suited to capture product functionality, and when applied properly, nonfunctional requirements. But user interface design and complex user interactions are better described by other techniques including design sketches, mock-ups, scenarios, and storyboards. Complement your user stories therefore with other techniques, and don't feel obliged to only use stories.

User Incognito

A user story tells a story about a user interacting with the product. Some stories, however, omit the beneficiary altogether, or they talk about a mysterious user as in "As the user, I want to …" But if it's not clear who the user is, why should the story be developed? How can we be confident that the story will benefit someone?

Make sure that all your stories have a clear user or customer. As I like to work with personas, I use personas in my stories (instead of user roles). This connects each story to the right persona, and it allows me to understand if and to which extend the story addresses the persona's need or goal. To achieve this, I use the following template: As <persona>, I want <something> so that <benefit>.

Disastrous Details

The devil is in the details: Some stories are too big and vague for the team to understand and implement. Others contain too much detail, or even prescribe a solution. Getting the details right seems to be a battle the product owner can only loose.

My recommendation is: Start with big, coarse-grained stories called epics particularly for all new features. Epics allow you to capture an idea without committing to the details. Then break an epic into more detailed user stories. The new user stories replace the epic, and they provide more information about the product's functionality. Pay particular attention to the user stories that are pulled into a sprint. These stories have to be ready: clear, feasible, and testable. (You can learn more about decomposing epics into ready stories in my post "Epics and Ready Stories".)

Progressively refining your stories keeps your Product Canvas or backlog concise, it makes it easier to integrate new insights, and it reduces the effort required to stock your initial canvas or backlog. This is particularly valuable for new products and new features. Make sure you do not prescribe a solution in your stories. Rather focus on the "what", nor the "how". The latter is best captured in an architecture diagram.

Story Handoff

Some product owners diligently write user stories, and give them to the development team in the sprint planning meeting. This handoff is usually suboptimal, as it wastes the team's ideas and knowledge. Stories can hence be inappropriate, difficult to understand, unfeasible and not testable.

User stories, however, are not meant to be standalone documents. They should be complemented by a conversation between the product owner and the team, or even better: written collaboratively. A story wants to capture the essentials, and not specify every detail: The latter would be difficult for new features and too slow and too expensive in an agile / lean context.

User story writing should be a team effort, where product owner and development team create the stories together. Allocate time for a collaborative Product Canvas workshop or backlog grooming meeting, particularly when you develop a new product or new features. Trust me: Better stories and a better product will be your reward.

Criteria Crisis

Acceptance criteria are maybe the most misunderstood part of users stories. I have seen detailed stories with no acceptance criteria, criteria that restate the narrative, and criteria that hide new stories, or even contain entire workflows. The last two mistakes are exemplified by the following story (the narrative is on the card on the left, the "acceptance criteria" on the right):

The idea behind acceptance criteria is simple: They allow you to describe the conditions that have to be fulfilled so that the story is done, that it can be exposed to the users and the other stakeholders. This ensures that you gather feedback and/or release features, and it helps the team plan and track their work: The criteria enrich the story and make it more precise and testable. (The criteria all stories have to fulfil such as "online help is available" are not stated in the acceptance criteria but in the Definition of Done.)

As a rule of thumb, I like to work with three to five criteria per story, and I am not worried if my epics don't have acceptance criteria to start with. Ready stories, however, must provide meaningful criteria. You can find sample acceptance criteria in my posts "Epics and Ready Stories" and "Nonfunctional Requirements".

http://www.romanpichler.com/blog/user-stories/writing-good-user-stories/

10 Tips for Writing Good User Stories

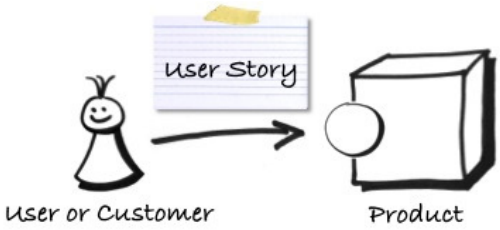
Posted on Thursday 7th October 2010

Summary

User stories are probably the most popular agile technique to capture product functionality: Working with user stories is easy. But writing good stories can be hard. The following ten tips help you create good stories.

1. Focus on the user

As its name suggests, a user story tells a story about a customer or user employing the product. Write therefore stories from the user's perspective and show how a user or customer uses some product functionality.

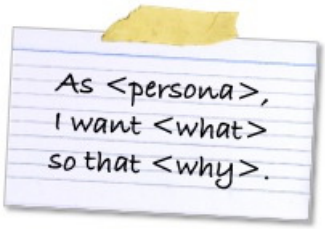


2. Write stories collaboratively

A user story is not a specification, but an communication and collaboration tool. Stories should not be handed off to the development team but be complemented by a conversation: Product owner and team should talk about stories, or even better, write them together: Invite the team to create and detail the stories together. This leverages the creativity and the knowledge of the team and usually results in better user stories.

3. Keep your stories simple and concise

Write your stories so that they are easy to understand. Keep them simple and concise. Avoid confusing and ambiguous terms, and use active voice. Focus on what's important, and leave out non-essential information. The following template puts the user or customer into the story and makes its benefit explicit. But use the template only if it is helpful. Try out different ways to write your stories to understand what works best for you.

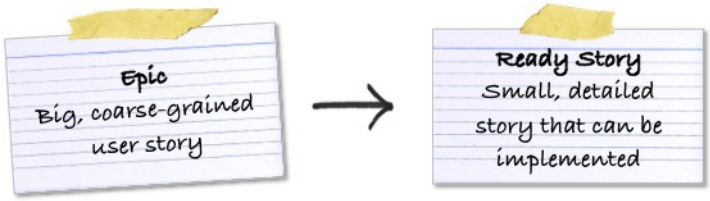


4. Start with epics

Epics are big, coarse-grained user stories. Starting with epics allows you to sketch the product functionality without committing to the details. This is particularly helpful for new products and new features, as it buys you time to learn more about the users and how to best meet their needs. It also reduces the time and effort required to integrate new insights: If you have lots of detailed stories, then it's often tricky to relate the feedback you receive to the right stories. I like to use the personas with their goals to discover the right epics.

5. Decompose your stories until they are ready

Break your epics into smaller, detailed stories until they are ready: clear, feasible, and testable. Everyone should have a shared understanding of the story's meaning; the story should not too big, and there has to be an effective way to determine if the story is done.



6. Add acceptance criteria

As you decompose epics into smaller stories, remember to add acceptance criteria. Acceptance criteria complement the story's narrative: They allow you to describe the conditions that have to be fulfilled so that the story is done. The criteria enrich the story and make it more precise and testable, and they ensures that the story can be demoed or released to the users and the other stakeholders.

7. Group user stories into themes

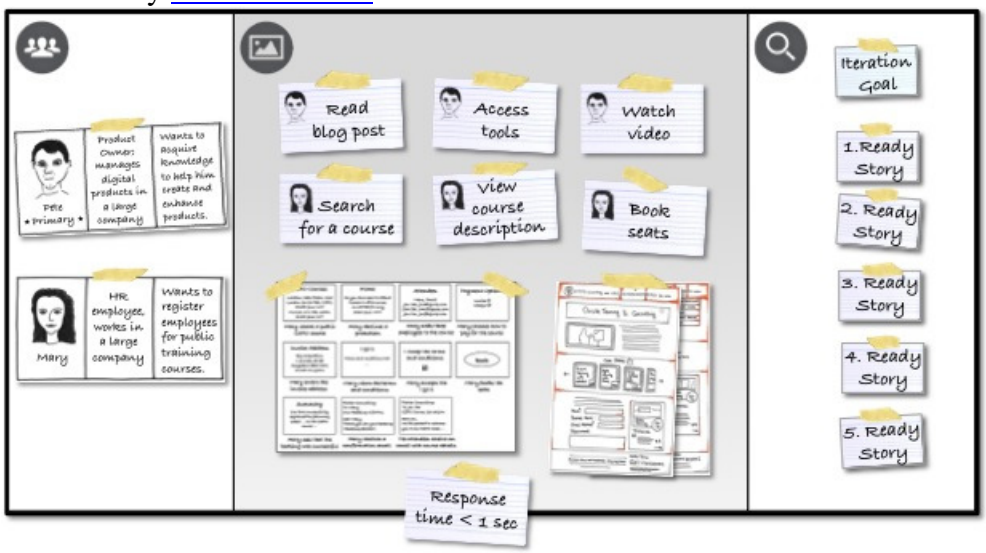
Using themes helps you organise your stories. Each theme is a group of related stories. Sample themes for a mobile phone are email, calendar, voice communication, and organizer, for instance. Themes make it easier to check for completeness and consistency, they structure your product canvas or product backlog, and they facilitate prioritisation.

8. Use paper cards

Paper cards are not only cheap and easy to use. They facilitate collaboration, as everyone can take a card and jot down an idea. Cards can also be easily grouped on the table or wall to check for consistency and completeness. Even if your stories are stored electronically, it is worthwhile to use paper cards when you write new stories.

9. Keep your stories visible and accessible

Stories want to communication information. Don't hide them on a network drive, the corporate intranet jungle, or a licensed tool. Make them visible instead, for instance, by putting them up on the wall. A great tool to discover, visualise, and manage your stories is my Product Canvas.



10. Don't solely rely on user stories

Creating a great user experience (UX) requires more than writing user stories. Also consider the user journeys and interactions, the visual design, and the nonfunctional properties of your product. This results in a holistic description that makes it easier to identify risks and assumptions, and it increases the chances of creating a product with the right user experience.

http://www.romanpichler.com/blog/user-stories/5-common-user-story-mistakes/

5 Common User Story Mistakes

Posted on Wednesday 12th June 2013

Summary

This post discusses five common user story mistakes and helps you overcome them.

User stories are a simple, yet effective way to communicate how a user or customer employs a product. But writing user stories that help a team build great software can be challenging. The following list summarises five common user story mistakes to help you improve your stories.

Story Mania

Some product owners and teams are so fond of user stories that everything is expressed as a story. This either results in some rather odd stories – stories that capture the user interface design, complex user interactions, and technical requirements; or these aspects are simply overlooked.

Like any technique, user story writing has its strengths and limitations. I find stories particularly well suited to capture product functionality, and when applied properly, nonfunctional requirements. But user interface design and complex user interactions are better described by other techniques including design sketches, mock-ups, scenarios, and storyboards. Complement your user stories therefore with other techniques, and don't feel obliged to only use stories.

User Incognito

A user story tells a story about a user interacting with the product. Some stories, however, omit the beneficiary altogether, or they talk about a mysterious user as in "As the user, I want to …" But if it's not clear who the user is, why should the story be developed? How can we be confident that the story will benefit someone?

Make sure that all your stories have a clear user or customer. As I like to work with personas, I use personas in my stories (instead of user roles). This connects each story to the right persona, and it allows me to understand if and to which extend the story addresses the persona's need or goal. To achieve this, I use the following template: As <persona>, I want <something> so that <benefit>.

Disastrous Details

The devil is in the details: Some stories are too big and vague for the team to understand and implement. Others contain too much detail, or even prescribe a solution. Getting the details right seems to be a battle the product owner can only loose.

My recommendation is: Start with big, coarse-grained stories called epics particularly for all new features. Epics allow you to capture an idea without committing to the details. Then break an epic into more detailed user stories. The new user stories replace the epic, and they provide more information about the product's functionality. Pay particular attention to the user stories that are pulled into a sprint. These stories have to be ready: clear, feasible, and testable. (You can learn more about decomposing epics into ready stories in my post "Epics and Ready Stories".)

Progressively refining your stories keeps your Product Canvas or backlog concise, it makes it easier to integrate new insights, and it reduces the effort required to stock your initial canvas or backlog. This is particularly valuable for new products and new features. Make sure you do not prescribe a solution in your stories. Rather focus on the "what", nor the "how". The latter is best captured in an architecture diagram.

Story Handoff

Some product owners diligently write user stories, and give them to the development team in the sprint planning meeting. This handoff is usually suboptimal, as it wastes the team's ideas and knowledge. Stories can hence be inappropriate, difficult to understand, unfeasible and not testable.

User stories, however, are not meant to be standalone documents. They should be complemented by a conversation between the product owner and the team, or even better: written collaboratively. A story wants to capture the essentials, and not specify every detail: The latter would be difficult for new features and too slow and too expensive in an agile / lean context.

User story writing should be a team effort, where product owner and development team create the stories together. Allocate time for a collaborative Product Canvas workshop or backlog grooming meeting, particularly when you develop a new product or new features. Trust me: Better stories and a better product will be your reward.

Criteria Crisis

Acceptance criteria are maybe the most misunderstood part of users stories. I have seen detailed stories with no acceptance criteria, criteria that restate the narrative, and criteria that hide new stories, or even contain entire workflows. The last two mistakes are exemplified by the following story (the narrative is on the card on the left, the "acceptance criteria" on the right):

The idea behind acceptance criteria is simple: They allow you to describe the conditions that have to be fulfilled so that the story is done, that it can be exposed to the users and the other stakeholders. This ensures that you gather feedback and/or release features, and it helps the team plan and track their work: The criteria enrich the story and make it more precise and testable. (The criteria all stories have to fulfil such as "online help is available" are not stated in the acceptance criteria but in the Definition of Done.)

As a rule of thumb, I like to work with three to five criteria per story, and I am not worried if my epics don't have acceptance criteria to start with. Ready stories, however, must provide meaningful criteria. You can find sample acceptance criteria in my posts "Epics and Ready Stories" and "Nonfunctional Requirements".

http://www.romanpichler.com/blog/user-stories/writing-good-user-stories/

10 Tips for Writing Good User Stories

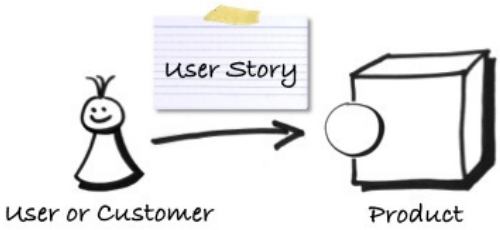
Posted on Thursday 7th October 2010

Summary

User stories are probably the most popular agile technique to capture product functionality: Working with user stories is easy. But writing good stories can be hard. The following ten tips help you create good stories.

1. Focus on the user

As its name suggests, a user story tells a story about a customer or user employing the product. Write therefore stories from the user's perspective and show how a user or customer uses some product functionality.

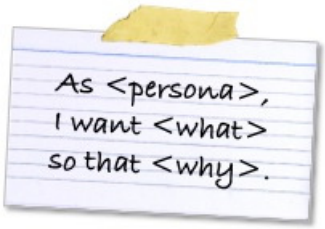


2. Write stories collaboratively

A user story is not a specification, but an communication and collaboration tool. Stories should not be handed off to the development team but be complemented by a conversation: Product owner and team should talk about stories, or even better, write them together: Invite the team to create and detail the stories together. This leverages the creativity and the knowledge of the team and usually results in better user stories.

3. Keep your stories simple and concise

Write your stories so that they are easy to understand. Keep them simple and concise. Avoid confusing and ambiguous terms, and use active voice. Focus on what's important, and leave out non-essential information. The following template puts the user or customer into the story and makes its benefit explicit. But use the template only if it is helpful. Try out different ways to write your stories to understand what works best for you.

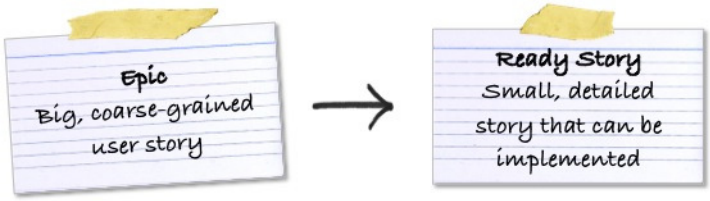


4. Start with epics

Epics are big, coarse-grained user stories. Starting with epics allows you to sketch the product functionality without committing to the details. This is particularly helpful for new products and new features, as it buys you time to learn more about the users and how to best meet their needs. It also reduces the time and effort required to integrate new insights: If you have lots of detailed stories, then it's often tricky to relate the feedback you receive to the right stories. I like to use the personas with their goals to discover the right epics.

5. Decompose your stories until they are ready

Break your epics into smaller, detailed stories until they are ready: clear, feasible, and testable. Everyone should have a shared understanding of the story's meaning; the story should not too big, and there has to be an effective way to determine if the story is done.



6. Add acceptance criteria

As you decompose epics into smaller stories, remember to add acceptance criteria. Acceptance criteria complement the story's narrative: They allow you to describe the conditions that have to be fulfilled so that the story is done. The criteria enrich the story and make it more precise and testable, and they ensures that the story can be demoed or released to the users and the other stakeholders.

7. Group user stories into themes

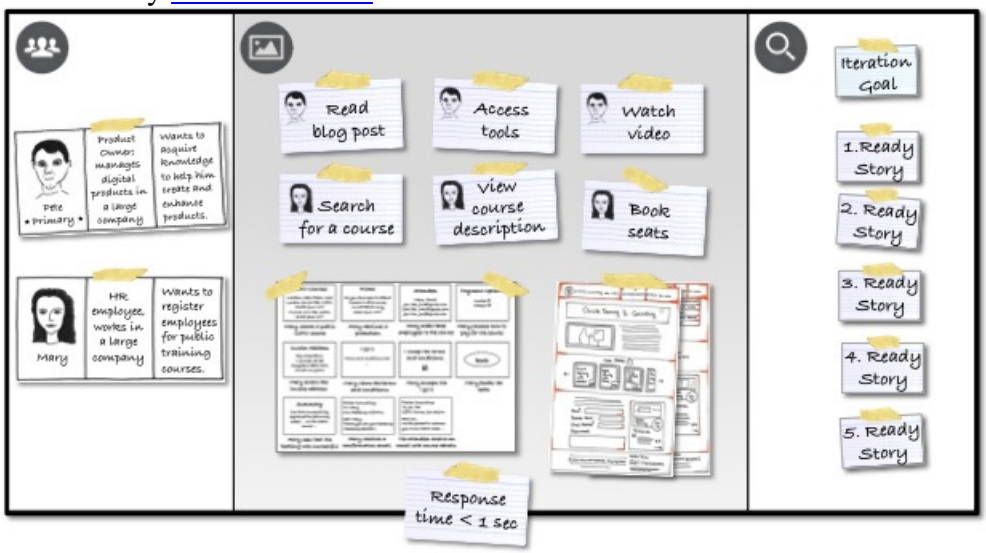
Using themes helps you organise your stories. Each theme is a group of related stories. Sample themes for a mobile phone are email, calendar, voice communication, and organizer, for instance. Themes make it easier to check for completeness and consistency, they structure your product canvas or product backlog, and they facilitate prioritisation.

8. Use paper cards

Paper cards are not only cheap and easy to use. They facilitate collaboration, as everyone can take a card and jot down an idea. Cards can also be easily grouped on the table or wall to check for consistency and completeness. Even if your stories are stored electronically, it is worthwhile to use paper cards when you write new stories.

9. Keep your stories visible and accessible

Stories want to communication information. Don't hide them on a network drive, the corporate intranet jungle, or a licensed tool. Make them visible instead, for instance, by putting them up on the wall. A great tool to discover, visualise, and manage your stories is my Product Canvas.



10. Don't solely rely on user stories

Creating a great user experience (UX) requires more than writing user stories. Also consider the user journeys and interactions, the visual design, and the nonfunctional properties of your product. This results in a holistic description that makes it easier to identify risks and assumptions, and it increases the chances of creating a product with the right user experience.

http://www.romanpichler.com/blog/user-stories/5-common-user-story-mistakes/

5 Common User Story Mistakes

Posted on Wednesday 12th June 2013

Summary

This post discusses five common user story mistakes and helps you overcome them.

User stories are a simple, yet effective way to communicate how a user or customer employs a product. But writing user stories that help a team build great software can be challenging. The following list summarises five common user story mistakes to help you improve your stories.

Story Mania

Some product owners and teams are so fond of user stories that everything is expressed as a story. This either results in some rather odd stories – stories that capture the user interface design, complex user interactions, and technical requirements; or these aspects are simply overlooked.

Like any technique, user story writing has its strengths and limitations. I find stories particularly well suited to capture product functionality, and when applied properly, nonfunctional requirements. But user interface design and complex user interactions are better described by other techniques including design sketches, mock-ups, scenarios, and storyboards. Complement your user stories therefore with other techniques, and don't feel obliged to only use stories.

User Incognito

A user story tells a story about a user interacting with the product. Some stories, however, omit the beneficiary altogether, or they talk about a mysterious user as in "As the user, I want to …" But if it's not clear who the user is, why should the story be developed? How can we be confident that the story will benefit someone?

Make sure that all your stories have a clear user or customer. As I like to work with personas, I use personas in my stories (instead of user roles). This connects each story to the right persona, and it allows me to understand if and to which extend the story addresses the persona's need or goal. To achieve this, I use the following template: As <persona>, I want <something> so that <benefit>.

Disastrous Details

The devil is in the details: Some stories are too big and vague for the team to understand and implement. Others contain too much detail, or even prescribe a solution. Getting the details right seems to be a battle the product owner can only loose.

My recommendation is: Start with big, coarse-grained stories called epics particularly for all new features. Epics allow you to capture an idea without committing to the details. Then break an epic into more detailed user stories. The new user stories replace the epic, and they provide more information about the product's functionality. Pay particular attention to the user stories that are pulled into a sprint. These stories have to be ready: clear, feasible, and testable. (You can learn more about decomposing epics into ready stories in my post "Epics and Ready Stories".)

Progressively refining your stories keeps your Product Canvas or backlog concise, it makes it easier to integrate new insights, and it reduces the effort required to stock your initial canvas or backlog. This is particularly valuable for new products and new features. Make sure you do not prescribe a solution in your stories. Rather focus on the "what", nor the "how". The latter is best captured in an architecture diagram.

Story Handoff

Some product owners diligently write user stories, and give them to the development team in the sprint planning meeting. This handoff is usually suboptimal, as it wastes the team's ideas and knowledge. Stories can hence be inappropriate, difficult to understand, unfeasible and not testable.

User stories, however, are not meant to be standalone documents. They should be complemented by a conversation between the product owner and the team, or even better: written collaboratively. A story wants to capture the essentials, and not specify every detail: The latter would be difficult for new features and too slow and too expensive in an agile / lean context.

User story writing should be a team effort, where product owner and development team create the stories together. Allocate time for a collaborative Product Canvas workshop or backlog grooming meeting, particularly when you develop a new product or new features. Trust me: Better stories and a better product will be your reward.

Criteria Crisis

Acceptance criteria are maybe the most misunderstood part of users stories. I have seen detailed stories with no acceptance criteria, criteria that restate the narrative, and criteria that hide new stories, or even contain entire workflows. The last two mistakes are exemplified by the following story (the narrative is on the card on the left, the "acceptance criteria" on the right):

The idea behind acceptance criteria is simple: They allow you to describe the conditions that have to be fulfilled so that the story is done, that it can be exposed to the users and the other stakeholders. This ensures that you gather feedback and/or release features, and it helps the team plan and track their work: The criteria enrich the story and make it more precise and testable. (The criteria all stories have to fulfil such as "online help is available" are not stated in the acceptance criteria but in the Definition of Done.)

As a rule of thumb, I like to work with three to five criteria per story, and I am not worried if my epics don't have acceptance criteria to start with. Ready stories, however, must provide meaningful criteria. You can find sample acceptance criteria in my posts "Epics and Ready Stories" and "Nonfunctional Requirements".

http://www.romanpichler.com/blog/user-stories/writing-good-user-stories/

10 Tips for Writing Good User Stories

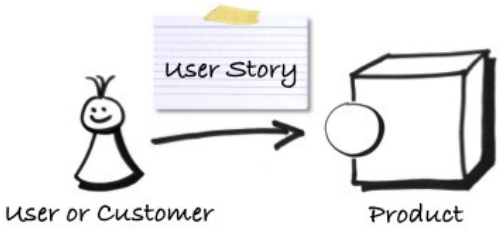
Posted on Thursday 7th October 2010

Summary

User stories are probably the most popular agile technique to capture product functionality: Working with user stories is easy. But writing good stories can be hard. The following ten tips help you create good stories.

1. Focus on the user

As its name suggests, a user story tells a story about a customer or user employing the product. Write therefore stories from the user's perspective and show how a user or customer uses some product functionality.

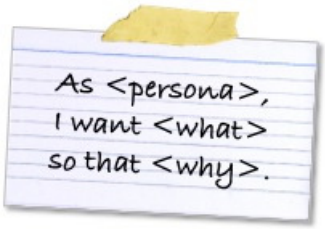


2. Write stories collaboratively

A user story is not a specification, but an communication and collaboration tool. Stories should not be handed off to the development team but be complemented by a conversation: Product owner and team should talk about stories, or even better, write them together: Invite the team to create and detail the stories together. This leverages the creativity and the knowledge of the team and usually results in better user stories.

3. Keep your stories simple and concise

Write your stories so that they are easy to understand. Keep them simple and concise. Avoid confusing and ambiguous terms, and use active voice. Focus on what's important, and leave out non-essential information. The following template puts the user or customer into the story and makes its benefit explicit. But use the template only if it is helpful. Try out different ways to write your stories to understand what works best for you.

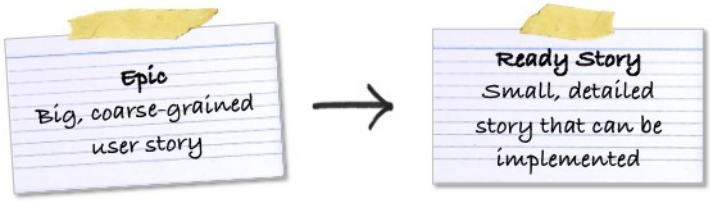


4. Start with epics

Epics are big, coarse-grained user stories. Starting with epics allows you to sketch the product functionality without committing to the details. This is particularly helpful for new products and new features, as it buys you time to learn more about the users and how to best meet their needs. It also reduces the time and effort required to integrate new insights: If you have lots of detailed stories, then it's often tricky to relate the feedback you receive to the right stories. I like to use the personas with their goals to discover the right epics.

5. Decompose your stories until they are ready

Break your epics into smaller, detailed stories until they are ready: clear, feasible, and testable. Everyone should have a shared understanding of the story's meaning; the story should not too big, and there has to be an effective way to determine if the story is done.



6. Add acceptance criteria

As you decompose epics into smaller stories, remember to add acceptance criteria. Acceptance criteria complement the story's narrative: They allow you to describe the conditions that have to be fulfilled so that the story is done. The criteria enrich the story and make it more precise and testable, and they ensures that the story can be demoed or released to the users and the other stakeholders.

7. Group user stories into themes

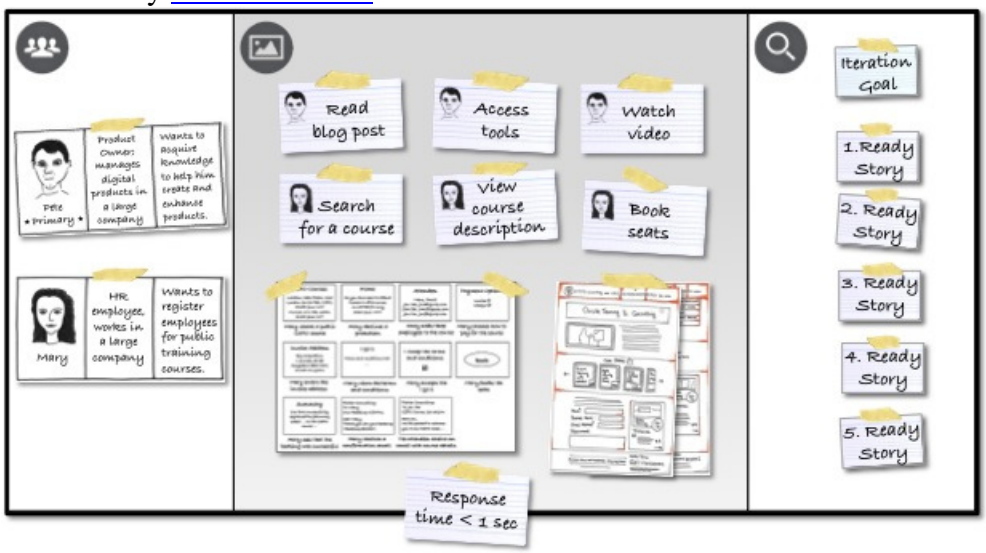
Using themes helps you organise your stories. Each theme is a group of related stories. Sample themes for a mobile phone are email, calendar, voice communication, and organizer, for instance. Themes make it easier to check for completeness and consistency, they structure your product canvas or product backlog, and they facilitate prioritisation.

8. Use paper cards

Paper cards are not only cheap and easy to use. They facilitate collaboration, as everyone can take a card and jot down an idea. Cards can also be easily grouped on the table or wall to check for consistency and completeness. Even if your stories are stored electronically, it is worthwhile to use paper cards when you write new stories.

9. Keep your stories visible and accessible

Stories want to communication information. Don't hide them on a network drive, the corporate intranet jungle, or a licensed tool. Make them visible instead, for instance, by putting them up on the wall. A great tool to discover, visualise, and manage your stories is my Product Canvas.



10. Don't solely rely on user stories

Creating a great user experience (UX) requires more than writing user stories. Also consider the user journeys and interactions, the visual design, and the nonfunctional properties of your product. This results in a holistic description that makes it easier to identify risks and assumptions, and it increases the chances of creating a product with the right user experience.

http://www.romanpichler.com/blog/user-stories/5-common-user-story-mistakes/

5 Common User Story Mistakes

Posted on Wednesday 12th June 2013

Summary

This post discusses five common user story mistakes and helps you overcome them.

User stories are a simple, yet effective way to communicate how a user or customer employs a product. But writing user stories that help a team build great software can be challenging. The following list summarises five common user story mistakes to help you improve your stories.

Story Mania

Some product owners and teams are so fond of user stories that everything is expressed as a story. This either results in some rather odd stories – stories that capture the user interface design, complex user interactions, and technical requirements; or these aspects are simply overlooked.

Like any technique, user story writing has its strengths and limitations. I find stories particularly well suited to capture product functionality, and when applied properly, nonfunctional requirements. But user interface design and complex user interactions are better described by other techniques including design sketches, mock-ups, scenarios, and storyboards. Complement your user stories therefore with other techniques, and don't feel obliged to only use stories.

User Incognito

A user story tells a story about a user interacting with the product. Some stories, however, omit the beneficiary altogether, or they talk about a mysterious user as in "As the user, I want to …" But if it's not clear who the user is, why should the story be developed? How can we be confident that the story will benefit someone?

Make sure that all your stories have a clear user or customer. As I like to work with personas, I use personas in my stories (instead of user roles). This connects each story to the right persona, and it allows me to understand if and to which extend the story addresses the persona's need or goal. To achieve this, I use the following template: As <persona>, I want <something> so that <benefit>.

Disastrous Details

The devil is in the details: Some stories are too big and vague for the team to understand and implement. Others contain too much detail, or even prescribe a solution. Getting the details right seems to be a battle the product owner can only loose.

My recommendation is: Start with big, coarse-grained stories called epics particularly for all new features. Epics allow you to capture an idea without committing to the details. Then break an epic into more detailed user stories. The new user stories replace the epic, and they provide more information about the product's functionality. Pay particular attention to the user stories that are pulled into a sprint. These stories have to be ready: clear, feasible, and testable. (You can learn more about decomposing epics into ready stories in my post "Epics and Ready Stories".)

Progressively refining your stories keeps your Product Canvas or backlog concise, it makes it easier to integrate new insights, and it reduces the effort required to stock your initial canvas or backlog. This is particularly valuable for new products and new features. Make sure you do not prescribe a solution in your stories. Rather focus on the "what", nor the "how". The latter is best captured in an architecture diagram.

Story Handoff

Some product owners diligently write user stories, and give them to the development team in the sprint planning meeting. This handoff is usually suboptimal, as it wastes the team's ideas and knowledge. Stories can hence be inappropriate, difficult to understand, unfeasible and not testable.

User stories, however, are not meant to be standalone documents. They should be complemented by a conversation between the product owner and the team, or even better: written collaboratively. A story wants to capture the essentials, and not specify every detail: The latter would be difficult for new features and too slow and too expensive in an agile / lean context.

User story writing should be a team effort, where product owner and development team create the stories together. Allocate time for a collaborative Product Canvas workshop or backlog grooming meeting, particularly when you develop a new product or new features. Trust me: Better stories and a better product will be your reward.

Criteria Crisis

Acceptance criteria are maybe the most misunderstood part of users stories. I have seen detailed stories with no acceptance criteria, criteria that restate the narrative, and criteria that hide new stories, or even contain entire workflows. The last two mistakes are exemplified by the following story (the narrative is on the card on the left, the "acceptance criteria" on the right):

The idea behind acceptance criteria is simple: They allow you to describe the conditions that have to be fulfilled so that the story is done, that it can be exposed to the users and the other stakeholders. This ensures that you gather feedback and/or release features, and it helps the team plan and track their work: The criteria enrich the story and make it more precise and testable. (The criteria all stories have to fulfil such as "online help is available" are not stated in the acceptance criteria but in the Definition of Done.)

As a rule of thumb, I like to work with three to five criteria per story, and I am not worried if my epics don't have acceptance criteria to start with. Ready stories, however, must provide meaningful criteria. You can find sample acceptance criteria in my posts "Epics and Ready Stories" and "Nonfunctional Requirements".

http://www.romanpichler.com/blog/user-stories/writing-good-user-stories/

10 Tips for Writing Good User Stories

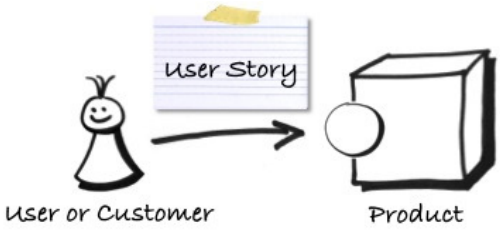
Posted on Thursday 7th October 2010

Summary

User stories are probably the most popular agile technique to capture product functionality: Working with user stories is easy. But writing good stories can be hard. The following ten tips help you create good stories.

1. Focus on the user

As its name suggests, a user story tells a story about a customer or user employing the product. Write therefore stories from the user's perspective and show how a user or customer uses some product functionality.

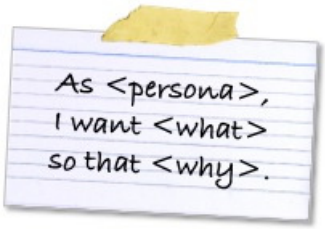


2. Write stories collaboratively

A user story is not a specification, but an communication and collaboration tool. Stories should not be handed off to the development team but be complemented by a conversation: Product owner and team should talk about stories, or even better, write them together: Invite the team to create and detail the stories together. This leverages the creativity and the knowledge of the team and usually results in better user stories.

3. Keep your stories simple and concise

Write your stories so that they are easy to understand. Keep them simple and concise. Avoid confusing and ambiguous terms, and use active voice. Focus on what's important, and leave out non-essential information. The following template puts the user or customer into the story and makes its benefit explicit. But use the template only if it is helpful. Try out different ways to write your stories to understand what works best for you.

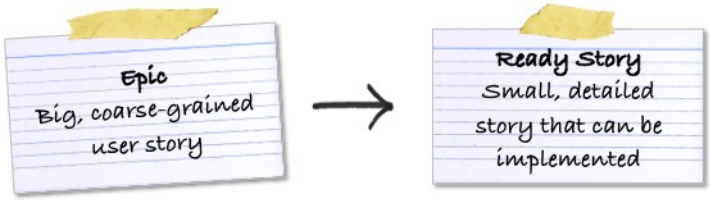


4. Start with epics

Epics are big, coarse-grained user stories. Starting with epics allows you to sketch the product functionality without committing to the details. This is particularly helpful for new products and new features, as it buys you time to learn more about the users and how to best meet their needs. It also reduces the time and effort required to integrate new insights: If you have lots of detailed stories, then it's often tricky to relate the feedback you receive to the right stories. I like to use the personas with their goals to discover the right epics.

5. Decompose your stories until they are ready

Break your epics into smaller, detailed stories until they are ready: clear, feasible, and testable. Everyone should have a shared understanding of the story's meaning; the story should not too big, and there has to be an effective way to determine if the story is done.



6. Add acceptance criteria

As you decompose epics into smaller stories, remember to add acceptance criteria. Acceptance criteria complement the story's narrative: They allow you to describe the conditions that have to be fulfilled so that the story is done. The criteria enrich the story and make it more precise and testable, and they ensures that the story can be demoed or released to the users and the other stakeholders.

7. Group user stories into themes

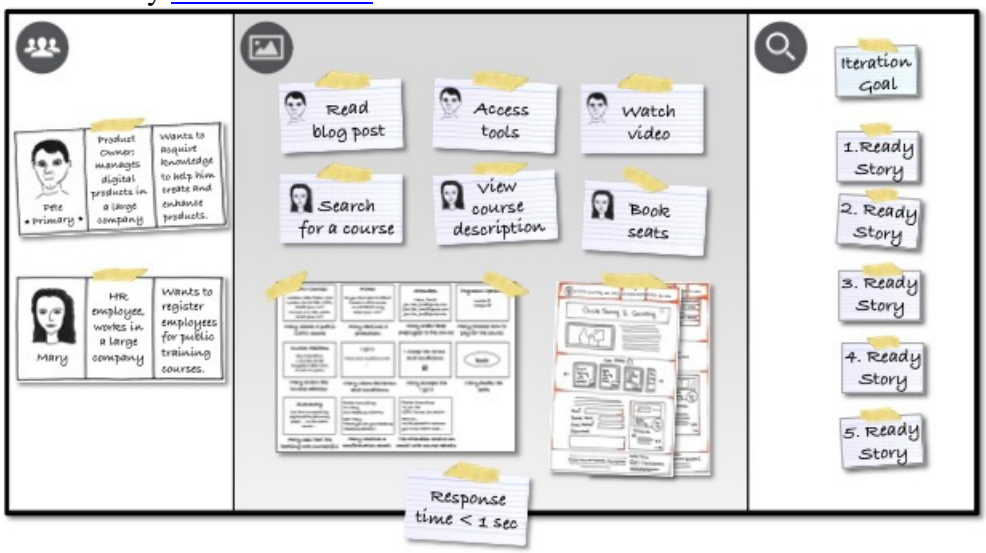
Using themes helps you organise your stories. Each theme is a group of related stories. Sample themes for a mobile phone are email, calendar, voice communication, and organizer, for instance. Themes make it easier to check for completeness and consistency, they structure your product canvas or product backlog, and they facilitate prioritisation.

8. Use paper cards

Paper cards are not only cheap and easy to use. They facilitate collaboration, as everyone can take a card and jot down an idea. Cards can also be easily grouped on the table or wall to check for consistency and completeness. Even if your stories are stored electronically, it is worthwhile to use paper cards when you write new stories.

9. Keep your stories visible and accessible

Stories want to communication information. Don't hide them on a network drive, the corporate intranet jungle, or a licensed tool. Make them visible instead, for instance, by putting them up on the wall. A great tool to discover, visualise, and manage your stories is my Product Canvas.



10. Don't solely rely on user stories

Creating a great user experience (UX) requires more than writing user stories. Also consider the user journeys and interactions, the visual design, and the nonfunctional properties of your product. This results in a holistic description that makes it easier to identify risks and assumptions, and it increases the chances of creating a product with the right user experience.

http://www.romanpichler.com/blog/user-stories/5-common-user-story-mistakes/

5 Common User Story Mistakes

Posted on Wednesday 12th June 2013

Summary

This post discusses five common user story mistakes and helps you overcome them.

User stories are a simple, yet effective way to communicate how a user or customer employs a product. But writing user stories that help a team build great software can be challenging. The following list summarises five common user story mistakes to help you improve your stories.

Story Mania

Some product owners and teams are so fond of user stories that everything is expressed as a story. This either results in some rather odd stories – stories that capture the user interface design, complex user interactions, and technical requirements; or these aspects are simply overlooked.

Like any technique, user story writing has its strengths and limitations. I find stories particularly well suited to capture product functionality, and when applied properly, nonfunctional requirements. But user interface design and complex user interactions are better described by other techniques including design sketches, mock-ups, scenarios, and storyboards. Complement your user stories therefore with other techniques, and don't feel obliged to only use stories.

User Incognito

A user story tells a story about a user interacting with the product. Some stories, however, omit the beneficiary altogether, or they talk about a mysterious user as in "As the user, I want to …" But if it's not clear who the user is, why should the story be developed? How can we be confident that the story will benefit someone?

Make sure that all your stories have a clear user or customer. As I like to work with personas, I use personas in my stories (instead of user roles). This connects each story to the right persona, and it allows me to understand if and to which extend the story addresses the persona's need or goal. To achieve this, I use the following template: As <persona>, I want <something> so that <benefit>.

Disastrous Details

The devil is in the details: Some stories are too big and vague for the team to understand and implement. Others contain too much detail, or even prescribe a solution. Getting the details right seems to be a battle the product owner can only loose.

My recommendation is: Start with big, coarse-grained stories called epics particularly for all new features. Epics allow you to capture an idea without committing to the details. Then break an epic into more detailed user stories. The new user stories replace the epic, and they provide more information about the product's functionality. Pay particular attention to the user stories that are pulled into a sprint. These stories have to be ready: clear, feasible, and testable. (You can learn more about decomposing epics into ready stories in my post "Epics and Ready Stories".)

Progressively refining your stories keeps your Product Canvas or backlog concise, it makes it easier to integrate new insights, and it reduces the effort required to stock your initial canvas or backlog. This is particularly valuable for new products and new features. Make sure you do not prescribe a solution in your stories. Rather focus on the "what", nor the "how". The latter is best captured in an architecture diagram.

Story Handoff

Some product owners diligently write user stories, and give them to the development team in the sprint planning meeting. This handoff is usually suboptimal, as it wastes the team's ideas and knowledge. Stories can hence be inappropriate, difficult to understand, unfeasible and not testable.

User stories, however, are not meant to be standalone documents. They should be complemented by a conversation between the product owner and the team, or even better: written collaboratively. A story wants to capture the essentials, and not specify every detail: The latter would be difficult for new features and too slow and too expensive in an agile / lean context.

User story writing should be a team effort, where product owner and development team create the stories together. Allocate time for a collaborative Product Canvas workshop or backlog grooming meeting, particularly when you develop a new product or new features. Trust me: Better stories and a better product will be your reward.

Criteria Crisis

Acceptance criteria are maybe the most misunderstood part of users stories. I have seen detailed stories with no acceptance criteria, criteria that restate the narrative, and criteria that hide new stories, or even contain entire workflows. The last two mistakes are exemplified by the following story (the narrative is on the card on the left, the "acceptance criteria" on the right):

The idea behind acceptance criteria is simple: They allow you to describe the conditions that have to be fulfilled so that the story is done, that it can be exposed to the users and the other stakeholders. This ensures that you gather feedback and/or release features, and it helps the team plan and track their work: The criteria enrich the story and make it more precise and testable. (The criteria all stories have to fulfil such as "online help is available" are not stated in the acceptance criteria but in the Definition of Done.)

As a rule of thumb, I like to work with three to five criteria per story, and I am not worried if my epics don't have acceptance criteria to start with. Ready stories, however, must provide meaningful criteria. You can find sample acceptance criteria in my posts "Epics and Ready Stories" and "Nonfunctional Requirements".

http://www.romanpichler.com/blog/user-stories/writing-good-user-stories/

10 Tips for Writing Good User Stories

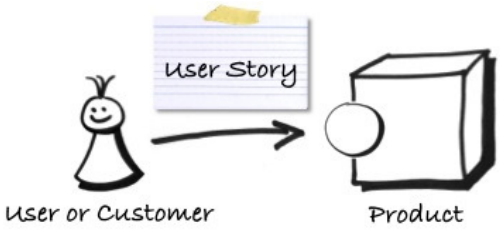
Posted on Thursday 7th October 2010

Summary

User stories are probably the most popular agile technique to capture product functionality: Working with user stories is easy. But writing good stories can be hard. The following ten tips help you create good stories.

1. Focus on the user

As its name suggests, a user story tells a story about a customer or user employing the product. Write therefore stories from the user's perspective and show how a user or customer uses some product functionality.

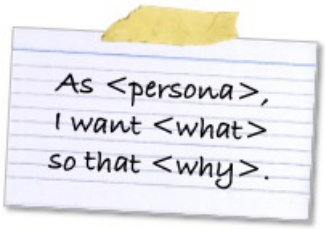


2. Write stories collaboratively

A user story is not a specification, but an communication and collaboration tool. Stories should not be handed off to the development team but be complemented by a conversation: Product owner and team should talk about stories, or even better, write them together: Invite the team to create and detail the stories together. This leverages the creativity and the knowledge of the team and usually results in better user stories.

3. Keep your stories simple and concise

Write your stories so that they are easy to understand. Keep them simple and concise. Avoid confusing and ambiguous terms, and use active voice. Focus on what's important, and leave out non-essential information. The following template puts the user or customer into the story and makes its benefit explicit. But use the template only if it is helpful. Try out different ways to write your stories to understand what works best for you.

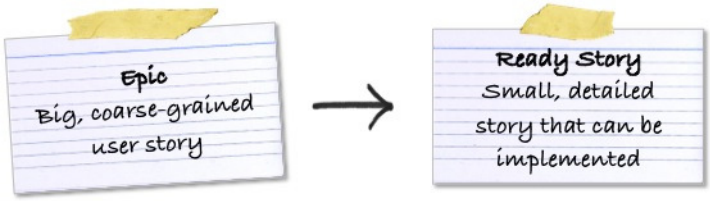


4. Start with epics

Epics are big, coarse-grained user stories. Starting with epics allows you to sketch the product functionality without committing to the details. This is particularly helpful for new products and new features, as it buys you time to learn more about the users and how to best meet their needs. It also reduces the time and effort required to integrate new insights: If you have lots of detailed stories, then it's often tricky to relate the feedback you receive to the right stories. I like to use the personas with their goals to discover the right epics.

5. Decompose your stories until they are ready

Break your epics into smaller, detailed stories until they are ready: clear, feasible, and testable. Everyone should have a shared understanding of the story's meaning; the story should not too big, and there has to be an effective way to determine if the story is done.



6. Add acceptance criteria

As you decompose epics into smaller stories, remember to add acceptance criteria. Acceptance criteria complement the story's narrative: They allow you to describe the conditions that have to be fulfilled so that the story is done. The criteria enrich the story and make it more precise and testable, and they ensures that the story can be demoed or released to the users and the other stakeholders.

7. Group user stories into themes

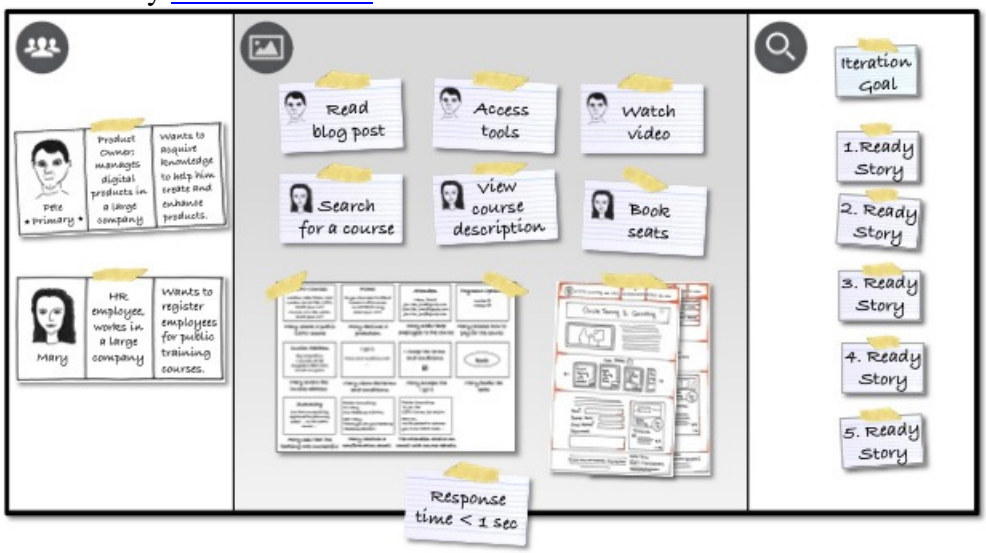
Using themes helps you organise your stories. Each theme is a group of related stories. Sample themes for a mobile phone are email, calendar, voice communication, and organizer, for instance. Themes make it easier to check for completeness and consistency, they structure your product canvas or product backlog, and they facilitate prioritisation.

8. Use paper cards

Paper cards are not only cheap and easy to use. They facilitate collaboration, as everyone can take a card and jot down an idea. Cards can also be easily grouped on the table or wall to check for consistency and completeness. Even if your stories are stored electronically, it is worthwhile to use paper cards when you write new stories.

9. Keep your stories visible and accessible

Stories want to communication information. Don't hide them on a network drive, the corporate intranet jungle, or a licensed tool. Make them visible instead, for instance, by putting them up on the wall. A great tool to discover, visualise, and manage your stories is my Product Canvas.



10. Don't solely rely on user stories

Creating a great user experience (UX) requires more than writing user stories. Also consider the user journeys and interactions, the visual design, and the nonfunctional properties of your product. This results in a holistic description that makes it easier to identify risks and assumptions, and it increases the chances of creating a product with the right user experience.

http://www.romanpichler.com/blog/user-stories/5-common-user-story-mistakes/

5 Common User Story Mistakes

Posted on Wednesday 12th June 2013

Summary

This post discusses five common user story mistakes and helps you overcome them.

User stories are a simple, yet effective way to communicate how a user or customer employs a product. But writing user stories that help a team build great software can be challenging. The following list summarises five common user story mistakes to help you improve your stories.

Story Mania

Some product owners and teams are so fond of user stories that everything is expressed as a story. This either results in some rather odd stories – stories that capture the user interface design, complex user interactions, and technical requirements; or these aspects are simply overlooked.

Like any technique, user story writing has its strengths and limitations. I find stories particularly well suited to capture product functionality, and when applied properly, nonfunctional requirements. But user interface design and complex user interactions are better described by other techniques including design sketches, mock-ups, scenarios, and storyboards. Complement your user stories therefore with other techniques, and don't feel obliged to only use stories.

User Incognito

A user story tells a story about a user interacting with the product. Some stories, however, omit the beneficiary altogether, or they talk about a mysterious user as in "As the user, I want to …" But if it's not clear who the user is, why should the story be developed? How can we be confident that the story will benefit someone?

Make sure that all your stories have a clear user or customer. As I like to work with personas, I use personas in my stories (instead of user roles). This connects each story to the right persona, and it allows me to understand if and to which extend the story addresses the persona's need or goal. To achieve this, I use the following template: As <persona>, I want <something> so that <benefit>.

Disastrous Details

The devil is in the details: Some stories are too big and vague for the team to understand and implement. Others contain too much detail, or even prescribe a solution. Getting the details right seems to be a battle the product owner can only loose.

My recommendation is: Start with big, coarse-grained stories called epics particularly for all new features. Epics allow you to capture an idea without committing to the details. Then break an epic into more detailed user stories. The new user stories replace the epic, and they provide more information about the product's functionality. Pay particular attention to the user stories that are pulled into a sprint. These stories have to be ready: clear, feasible, and testable. (You can learn more about decomposing epics into ready stories in my post "Epics and Ready Stories".)

Progressively refining your stories keeps your Product Canvas or backlog concise, it makes it easier to integrate new insights, and it reduces the effort required to stock your initial canvas or backlog. This is particularly valuable for new products and new features. Make sure you do not prescribe a solution in your stories. Rather focus on the "what", nor the "how". The latter is best captured in an architecture diagram.

Story Handoff

Some product owners diligently write user stories, and give them to the development team in the sprint planning meeting. This handoff is usually suboptimal, as it wastes the team's ideas and knowledge. Stories can hence be inappropriate, difficult to understand, unfeasible and not testable.

User stories, however, are not meant to be standalone documents. They should be complemented by a conversation between the product owner and the team, or even better: written collaboratively. A story wants to capture the essentials, and not specify every detail: The latter would be difficult for new features and too slow and too expensive in an agile / lean context.

User story writing should be a team effort, where product owner and development team create the stories together. Allocate time for a collaborative Product Canvas workshop or backlog grooming meeting, particularly when you develop a new product or new features. Trust me: Better stories and a better product will be your reward.

Criteria Crisis

Acceptance criteria are maybe the most misunderstood part of users stories. I have seen detailed stories with no acceptance criteria, criteria that restate the narrative, and criteria that hide new stories, or even contain entire workflows. The last two mistakes are exemplified by the following story (the narrative is on the card on the left, the "acceptance criteria" on the right):

The idea behind acceptance criteria is simple: They allow you to describe the conditions that have to be fulfilled so that the story is done, that it can be exposed to the users and the other stakeholders. This ensures that you gather feedback and/or release features, and it helps the team plan and track their work: The criteria enrich the story and make it more precise and testable. (The criteria all stories have to fulfil such as "online help is available" are not stated in the acceptance criteria but in the Definition of Done.)

As a rule of thumb, I like to work with three to five criteria per story, and I am not worried if my epics don't have acceptance criteria to start with. Ready stories, however, must provide meaningful criteria. You can find sample acceptance criteria in my posts "Epics and Ready Stories" and "Nonfunctional Requirements".

http://www.romanpichler.com/blog/user-stories/writing-good-user-stories/

10 Tips for Writing Good User Stories

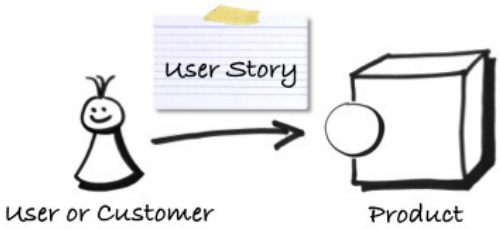
Posted on Thursday 7th October 2010

Summary

User stories are probably the most popular agile technique to capture product functionality: Working with user stories is easy. But writing good stories can be hard. The following ten tips help you create good stories.

1. Focus on the user

As its name suggests, a user story tells a story about a customer or user employing the product. Write therefore stories from the user's perspective and show how a user or customer uses some product functionality.

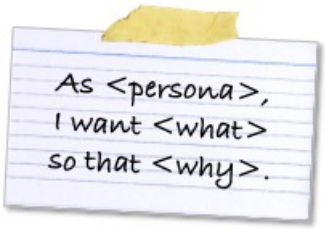


2. Write stories collaboratively

A user story is not a specification, but an communication and collaboration tool. Stories should not be handed off to the development team but be complemented by a conversation: Product owner and team should talk about stories, or even better, write them together: Invite the team to create and detail the stories together. This leverages the creativity and the knowledge of the team and usually results in better user stories.

3. Keep your stories simple and concise

Write your stories so that they are easy to understand. Keep them simple and concise. Avoid confusing and ambiguous terms, and use active voice. Focus on what's important, and leave out non-essential information. The following template puts the user or customer into the story and makes its benefit explicit. But use the template only if it is helpful. Try out different ways to write your stories to understand what works best for you.

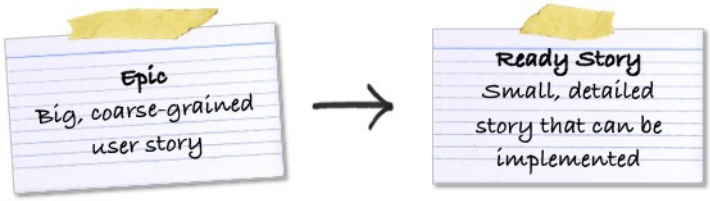


4. Start with epics

Epics are big, coarse-grained user stories. Starting with epics allows you to sketch the product functionality without committing to the details. This is particularly helpful for new products and new features, as it buys you time to learn more about the users and how to best meet their needs. It also reduces the time and effort required to integrate new insights: If you have lots of detailed stories, then it's often tricky to relate the feedback you receive to the right stories. I like to use the personas with their goals to discover the right epics.

5. Decompose your stories until they are ready

Break your epics into smaller, detailed stories until they are ready: clear, feasible, and testable. Everyone should have a shared understanding of the story's meaning; the story should not too big, and there has to be an effective way to determine if the story is done.



6. Add acceptance criteria

As you decompose epics into smaller stories, remember to add acceptance criteria. Acceptance criteria complement the story's narrative: They allow you to describe the conditions that have to be fulfilled so that the story is done. The criteria enrich the story and make it more precise and testable, and they ensures that the story can be demoed or released to the users and the other stakeholders.

7. Group user stories into themes

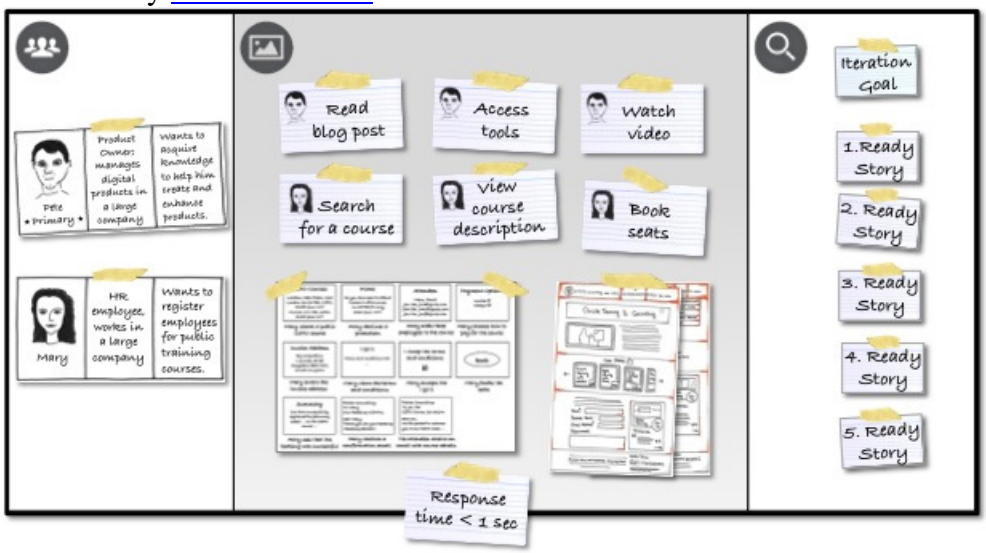
Using themes helps you organise your stories. Each theme is a group of related stories. Sample themes for a mobile phone are email, calendar, voice communication, and organizer, for instance. Themes make it easier to check for completeness and consistency, they structure your product canvas or product backlog, and they facilitate prioritisation.

8. Use paper cards

Paper cards are not only cheap and easy to use. They facilitate collaboration, as everyone can take a card and jot down an idea. Cards can also be easily grouped on the table or wall to check for consistency and completeness. Even if your stories are stored electronically, it is worthwhile to use paper cards when you write new stories.

9. Keep your stories visible and accessible

Stories want to communication information. Don't hide them on a network drive, the corporate intranet jungle, or a licensed tool. Make them visible instead, for instance, by putting them up on the wall. A great tool to discover, visualise, and manage your stories is my Product Canvas.



10. Don't solely rely on user stories

Creating a great user experience (UX) requires more than writing user stories. Also consider the user journeys and interactions, the visual design, and the nonfunctional properties of your product. This results in a holistic description that makes it easier to identify risks and assumptions, and it increases the chances of creating a product with the right user experience.

http://www.romanpichler.com/blog/user-stories/5-common-user-story-mistakes/

5 Common User Story Mistakes

Posted on Wednesday 12th June 2013

Summary

This post discusses five common user story mistakes and helps you overcome them.

User stories are a simple, yet effective way to communicate how a user or customer employs a product. But writing user stories that help a team build great software can be challenging. The following list summarises five common user story mistakes to help you improve your stories.

Story Mania

Some product owners and teams are so fond of user stories that everything is expressed as a story. This either results in some rather odd stories – stories that capture the user interface design, complex user interactions, and technical requirements; or these aspects are simply overlooked.

Like any technique, user story writing has its strengths and limitations. I find stories particularly well suited to capture product functionality, and when applied properly, nonfunctional requirements. But user interface design and complex user interactions are better described by other techniques including design sketches, mock-ups, scenarios, and storyboards. Complement your user stories therefore with other techniques, and don't feel obliged to only use stories.

User Incognito

A user story tells a story about a user interacting with the product. Some stories, however, omit the beneficiary altogether, or they talk about a mysterious user as in "As the user, I want to …" But if it's not clear who the user is, why should the story be developed? How can we be confident that the story will benefit someone?

Make sure that all your stories have a clear user or customer. As I like to work with personas, I use personas in my stories (instead of user roles). This connects each story to the right persona, and it allows me to understand if and to which extend the story addresses the persona's need or goal. To achieve this, I use the following template: As <persona>, I want <something> so that <benefit>.

Disastrous Details

The devil is in the details: Some stories are too big and vague for the team to understand and implement. Others contain too much detail, or even prescribe a solution. Getting the details right seems to be a battle the product owner can only loose.

My recommendation is: Start with big, coarse-grained stories called epics particularly for all new features. Epics allow you to capture an idea without committing to the details. Then break an epic into more detailed user stories. The new user stories replace the epic, and they provide more information about the product's functionality. Pay particular attention to the user stories that are pulled into a sprint. These stories have to be ready: clear, feasible, and testable. (You can learn more about decomposing epics into ready stories in my post "Epics and Ready Stories".)

Progressively refining your stories keeps your Product Canvas or backlog concise, it makes it easier to integrate new insights, and it reduces the effort required to stock your initial canvas or backlog. This is particularly valuable for new products and new features. Make sure you do not prescribe a solution in your stories. Rather focus on the "what", nor the "how". The latter is best captured in an architecture diagram.

Story Handoff

Some product owners diligently write user stories, and give them to the development team in the sprint planning meeting. This handoff is usually suboptimal, as it wastes the team's ideas and knowledge. Stories can hence be inappropriate, difficult to understand, unfeasible and not testable.

User stories, however, are not meant to be standalone documents. They should be complemented by a conversation between the product owner and the team, or even better: written collaboratively. A story wants to capture the essentials, and not specify every detail: The latter would be difficult for new features and too slow and too expensive in an agile / lean context.

User story writing should be a team effort, where product owner and development team create the stories together. Allocate time for a collaborative Product Canvas workshop or backlog grooming meeting, particularly when you develop a new product or new features. Trust me: Better stories and a better product will be your reward.

Criteria Crisis

Acceptance criteria are maybe the most misunderstood part of users stories. I have seen detailed stories with no acceptance criteria, criteria that restate the narrative, and criteria that hide new stories, or even contain entire workflows. The last two mistakes are exemplified by the following story (the narrative is on the card on the left, the "acceptance criteria" on the right):

The idea behind acceptance criteria is simple: They allow you to describe the conditions that have to be fulfilled so that the story is done, that it can be exposed to the users and the other stakeholders. This ensures that you gather feedback and/or release features, and it helps the team plan and track their work: The criteria enrich the story and make it more precise and testable. (The criteria all stories have to fulfil such as "online help is available" are not stated in the acceptance criteria but in the Definition of Done.)

As a rule of thumb, I like to work with three to five criteria per story, and I am not worried if my epics don't have acceptance criteria to start with. Ready stories, however, must provide meaningful criteria. You can find sample acceptance criteria in my posts "Epics and Ready Stories" and "Nonfunctional Requirements".

http://www.romanpichler.com/blog/user-stories/writing-good-user-stories/

10 Tips for Writing Good User Stories

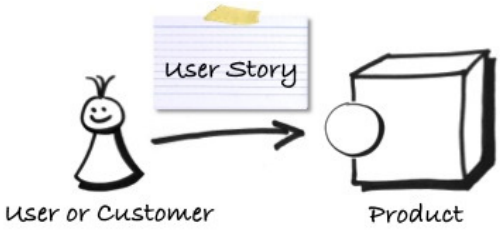
Posted on Thursday 7th October 2010

Summary

User stories are probably the most popular agile technique to capture product functionality: Working with user stories is easy. But writing good stories can be hard. The following ten tips help you create good stories.

1. Focus on the user

As its name suggests, a user story tells a story about a customer or user employing the product. Write therefore stories from the user's perspective and show how a user or customer uses some product functionality.

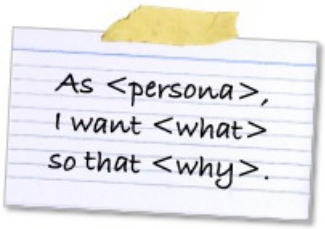


2. Write stories collaboratively

A user story is not a specification, but an communication and collaboration tool. Stories should not be handed off to the development team but be complemented by a conversation: Product owner and team should talk about stories, or even better, write them together: Invite the team to create and detail the stories together. This leverages the creativity and the knowledge of the team and usually results in better user stories.

3. Keep your stories simple and concise

Write your stories so that they are easy to understand. Keep them simple and concise. Avoid confusing and ambiguous terms, and use active voice. Focus on what's important, and leave out non-essential information. The following template puts the user or customer into the story and makes its benefit explicit. But use the template only if it is helpful. Try out different ways to write your stories to understand what works best for you.

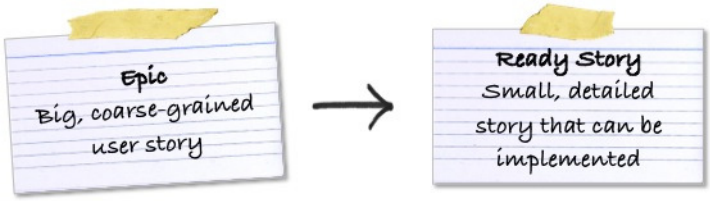


4. Start with epics

Epics are big, coarse-grained user stories. Starting with epics allows you to sketch the product functionality without committing to the details. This is particularly helpful for new products and new features, as it buys you time to learn more about the users and how to best meet their needs. It also reduces the time and effort required to integrate new insights: If you have lots of detailed stories, then it's often tricky to relate the feedback you receive to the right stories. I like to use the personas with their goals to discover the right epics.

5. Decompose your stories until they are ready

Break your epics into smaller, detailed stories until they are ready: clear, feasible, and testable. Everyone should have a shared understanding of the story's meaning; the story should not too big, and there has to be an effective way to determine if the story is done.



6. Add acceptance criteria

As you decompose epics into smaller stories, remember to add acceptance criteria. Acceptance criteria complement the story's narrative: They allow you to describe the conditions that have to be fulfilled so that the story is done. The criteria enrich the story and make it more precise and testable, and they ensures that the story can be demoed or released to the users and the other stakeholders.

7. Group user stories into themes

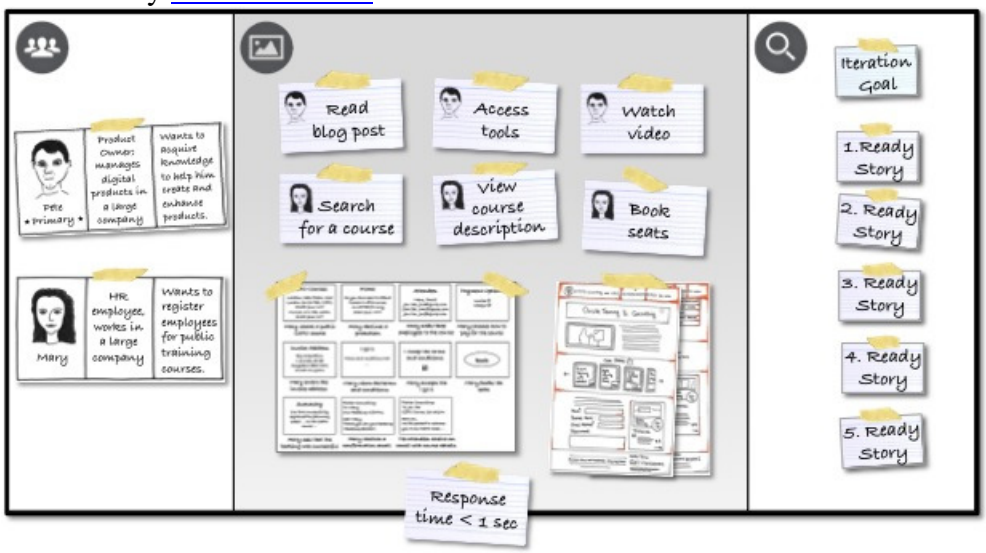
Using themes helps you organise your stories. Each theme is a group of related stories. Sample themes for a mobile phone are email, calendar, voice communication, and organizer, for instance. Themes make it easier to check for completeness and consistency, they structure your product canvas or product backlog, and they facilitate prioritisation.

8. Use paper cards

Paper cards are not only cheap and easy to use. They facilitate collaboration, as everyone can take a card and jot down an idea. Cards can also be easily grouped on the table or wall to check for consistency and completeness. Even if your stories are stored electronically, it is worthwhile to use paper cards when you write new stories.

9. Keep your stories visible and accessible

Stories want to communication information. Don't hide them on a network drive, the corporate intranet jungle, or a licensed tool. Make them visible instead, for instance, by putting them up on the wall. A great tool to discover, visualise, and manage your stories is my Product Canvas.



10. Don't solely rely on user stories

Creating a great user experience (UX) requires more than writing user stories. Also consider the user journeys and interactions, the visual design, and the nonfunctional properties of your product. This results in a holistic description that makes it easier to identify risks and assumptions, and it increases the chances of creating a product with the right user experience.

http://www.romanpichler.com/blog/user-stories/5-common-user-story-mistakes/

5 Common User Story Mistakes

Posted on Wednesday 12th June 2013

Summary

This post discusses five common user story mistakes and helps you overcome them.

User stories are a simple, yet effective way to communicate how a user or customer employs a product. But writing user stories that help a team build great software can be challenging. The following list summarises five common user story mistakes to help you improve your stories.

Story Mania

Some product owners and teams are so fond of user stories that everything is expressed as a story. This either results in some rather odd stories – stories that capture the user interface design, complex user interactions, and technical requirements; or these aspects are simply overlooked.

Like any technique, user story writing has its strengths and limitations. I find stories particularly well suited to capture product functionality, and when applied properly, nonfunctional requirements. But user interface design and complex user interactions are better described by other techniques including design sketches, mock-ups, scenarios, and storyboards. Complement your user stories therefore with other techniques, and don't feel obliged to only use stories.

User Incognito

A user story tells a story about a user interacting with the product. Some stories, however, omit the beneficiary altogether, or they talk about a mysterious user as in "As the user, I want to …" But if it's not clear who the user is, why should the story be developed? How can we be confident that the story will benefit someone?

Make sure that all your stories have a clear user or customer. As I like to work with personas, I use personas in my stories (instead of user roles). This connects each story to the right persona, and it allows me to understand if and to which extend the story addresses the persona's need or goal. To achieve this, I use the following template: As <persona>, I want <something> so that <benefit>.

Disastrous Details

The devil is in the details: Some stories are too big and vague for the team to understand and implement. Others contain too much detail, or even prescribe a solution. Getting the details right seems to be a battle the product owner can only loose.

My recommendation is: Start with big, coarse-grained stories called epics particularly for all new features. Epics allow you to capture an idea without committing to the details. Then break an epic into more detailed user stories. The new user stories replace the epic, and they provide more information about the product's functionality. Pay particular attention to the user stories that are pulled into a sprint. These stories have to be ready: clear, feasible, and testable. (You can learn more about decomposing epics into ready stories in my post "Epics and Ready Stories".)

Progressively refining your stories keeps your Product Canvas or backlog concise, it makes it easier to integrate new insights, and it reduces the effort required to stock your initial canvas or backlog. This is particularly valuable for new products and new features. Make sure you do not prescribe a solution in your stories. Rather focus on the "what", nor the "how". The latter is best captured in an architecture diagram.

Story Handoff

Some product owners diligently write user stories, and give them to the development team in the sprint planning meeting. This handoff is usually suboptimal, as it wastes the team's ideas and knowledge. Stories can hence be inappropriate, difficult to understand, unfeasible and not testable.

User stories, however, are not meant to be standalone documents. They should be complemented by a conversation between the product owner and the team, or even better: written collaboratively. A story wants to capture the essentials, and not specify every detail: The latter would be difficult for new features and too slow and too expensive in an agile / lean context.

User story writing should be a team effort, where product owner and development team create the stories together. Allocate time for a collaborative Product Canvas workshop or backlog grooming meeting, particularly when you develop a new product or new features. Trust me: Better stories and a better product will be your reward.

Criteria Crisis

Acceptance criteria are maybe the most misunderstood part of users stories. I have seen detailed stories with no acceptance criteria, criteria that restate the narrative, and criteria that hide new stories, or even contain entire workflows. The last two mistakes are exemplified by the following story (the narrative is on the card on the left, the "acceptance criteria" on the right):

The idea behind acceptance criteria is simple: They allow you to describe the conditions that have to be fulfilled so that the story is done, that it can be exposed to the users and the other stakeholders. This ensures that you gather feedback and/or release features, and it helps the team plan and track their work: The criteria enrich the story and make it more precise and testable. (The criteria all stories have to fulfil such as "online help is available" are not stated in the acceptance criteria but in the Definition of Done.)

As a rule of thumb, I like to work with three to five criteria per story, and I am not worried if my epics don't have acceptance criteria to start with. Ready stories, however, must provide meaningful criteria. You can find sample acceptance criteria in my posts "Epics and Ready Stories" and "Nonfunctional Requirements".

http://www.romanpichler.com/blog/user-stories/writing-good-user-stories/

10 Tips for Writing Good User Stories

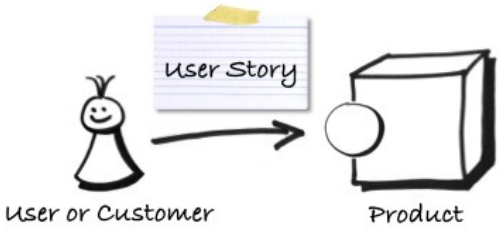
Posted on Thursday 7th October 2010

Summary

User stories are probably the most popular agile technique to capture product functionality: Working with user stories is easy. But writing good stories can be hard. The following ten tips help you create good stories.

1. Focus on the user

As its name suggests, a user story tells a story about a customer or user employing the product. Write therefore stories from the user's perspective and show how a user or customer uses some product functionality.

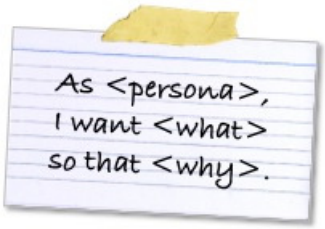


2. Write stories collaboratively

A user story is not a specification, but an communication and collaboration tool. Stories should not be handed off to the development team but be complemented by a conversation: Product owner and team should talk about stories, or even better, write them together: Invite the team to create and detail the stories together. This leverages the creativity and the knowledge of the team and usually results in better user stories.

3. Keep your stories simple and concise

Write your stories so that they are easy to understand. Keep them simple and concise. Avoid confusing and ambiguous terms, and use active voice. Focus on what's important, and leave out non-essential information. The following template puts the user or customer into the story and makes its benefit explicit. But use the template only if it is helpful. Try out different ways to write your stories to understand what works best for you.

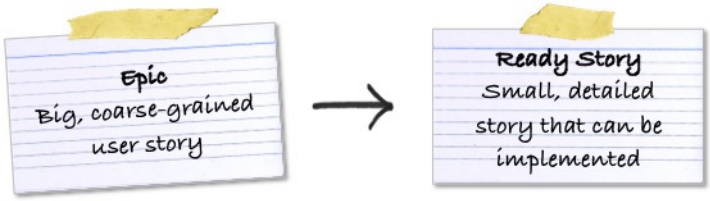


4. Start with epics

Epics are big, coarse-grained user stories. Starting with epics allows you to sketch the product functionality without committing to the details. This is particularly helpful for new products and new features, as it buys you time to learn more about the users and how to best meet their needs. It also reduces the time and effort required to integrate new insights: If you have lots of detailed stories, then it's often tricky to relate the feedback you receive to the right stories. I like to use the personas with their goals to discover the right epics.

5. Decompose your stories until they are ready

Break your epics into smaller, detailed stories until they are ready: clear, feasible, and testable. Everyone should have a shared understanding of the story's meaning; the story should not too big, and there has to be an effective way to determine if the story is done.



6. Add acceptance criteria

As you decompose epics into smaller stories, remember to add acceptance criteria. Acceptance criteria complement the story's narrative: They allow you to describe the conditions that have to be fulfilled so that the story is done. The criteria enrich the story and make it more precise and testable, and they ensures that the story can be demoed or released to the users and the other stakeholders.

7. Group user stories into themes

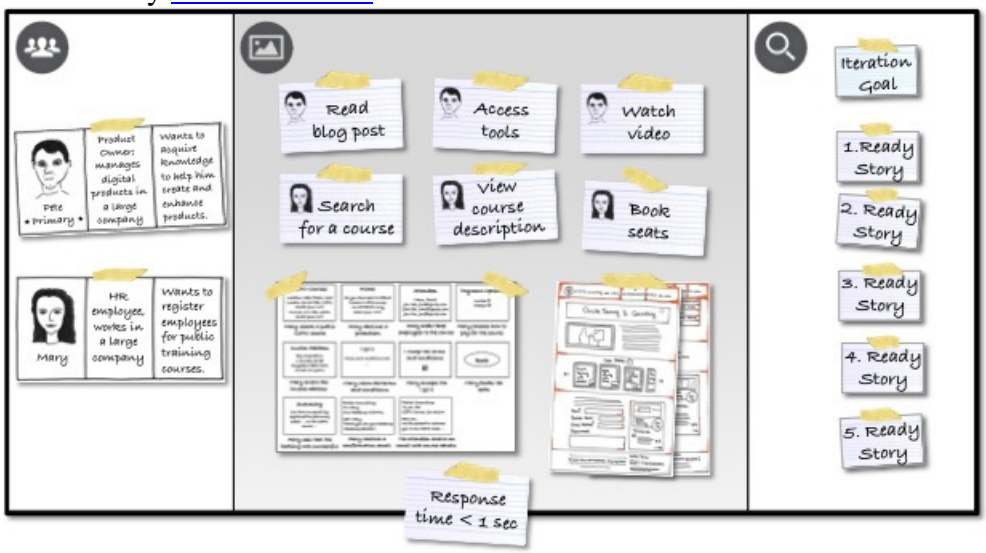
Using themes helps you organise your stories. Each theme is a group of related stories. Sample themes for a mobile phone are email, calendar, voice communication, and organizer, for instance. Themes make it easier to check for completeness and consistency, they structure your product canvas or product backlog, and they facilitate prioritisation.

8. Use paper cards

Paper cards are not only cheap and easy to use. They facilitate collaboration, as everyone can take a card and jot down an idea. Cards can also be easily grouped on the table or wall to check for consistency and completeness. Even if your stories are stored electronically, it is worthwhile to use paper cards when you write new stories.

9. Keep your stories visible and accessible

Stories want to communication information. Don't hide them on a network drive, the corporate intranet jungle, or a licensed tool. Make them visible instead, for instance, by putting them up on the wall. A great tool to discover, visualise, and manage your stories is my Product Canvas.



10. Don't solely rely on user stories

Creating a great user experience (UX) requires more than writing user stories. Also consider the user journeys and interactions, the visual design, and the nonfunctional properties of your product. This results in a holistic description that makes it easier to identify risks and assumptions, and it increases the chances of creating a product with the right user experience.

http://www.romanpichler.com/blog/user-stories/5-common-user-story-mistakes/

5 Common User Story Mistakes

Posted on Wednesday 12th June 2013

Summary

This post discusses five common user story mistakes and helps you overcome them.

User stories are a simple, yet effective way to communicate how a user or customer employs a product. But writing user stories that help a team build great software can be challenging. The following list summarises five common user story mistakes to help you improve your stories.

Story Mania

Some product owners and teams are so fond of user stories that everything is expressed as a story. This either results in some rather odd stories – stories that capture the user interface design, complex user interactions, and technical requirements; or these aspects are simply overlooked.

Like any technique, user story writing has its strengths and limitations. I find stories particularly well suited to capture product functionality, and when applied properly, nonfunctional requirements. But user interface design and complex user interactions are better described by other techniques including design sketches, mock-ups, scenarios, and storyboards. Complement your user stories therefore with other techniques, and don't feel obliged to only use stories.

User Incognito

A user story tells a story about a user interacting with the product. Some stories, however, omit the beneficiary altogether, or they talk about a mysterious user as in "As the user, I want to …" But if it's not clear who the user is, why should the story be developed? How can we be confident that the story will benefit someone?

Make sure that all your stories have a clear user or customer. As I like to work with personas, I use personas in my stories (instead of user roles). This connects each story to the right persona, and it allows me to understand if and to which extend the story addresses the persona's need or goal. To achieve this, I use the following template: As <persona>, I want <something> so that <benefit>.

Disastrous Details

The devil is in the details: Some stories are too big and vague for the team to understand and implement. Others contain too much detail, or even prescribe a solution. Getting the details right seems to be a battle the product owner can only loose.

My recommendation is: Start with big, coarse-grained stories called epics particularly for all new features. Epics allow you to capture an idea without committing to the details. Then break an epic into more detailed user stories. The new user stories replace the epic, and they provide more information about the product's functionality. Pay particular attention to the user stories that are pulled into a sprint. These stories have to be ready: clear, feasible, and testable. (You can learn more about decomposing epics into ready stories in my post "Epics and Ready Stories".)

Progressively refining your stories keeps your Product Canvas or backlog concise, it makes it easier to integrate new insights, and it reduces the effort required to stock your initial canvas or backlog. This is particularly valuable for new products and new features. Make sure you do not prescribe a solution in your stories. Rather focus on the "what", nor the "how". The latter is best captured in an architecture diagram.

Story Handoff

Some product owners diligently write user stories, and give them to the development team in the sprint planning meeting. This handoff is usually suboptimal, as it wastes the team's ideas and knowledge. Stories can hence be inappropriate, difficult to understand, unfeasible and not testable.

User stories, however, are not meant to be standalone documents. They should be complemented by a conversation between the product owner and the team, or even better: written collaboratively. A story wants to capture the essentials, and not specify every detail: The latter would be difficult for new features and too slow and too expensive in an agile / lean context.

User story writing should be a team effort, where product owner and development team create the stories together. Allocate time for a collaborative Product Canvas workshop or backlog grooming meeting, particularly when you develop a new product or new features. Trust me: Better stories and a better product will be your reward.

Criteria Crisis

Acceptance criteria are maybe the most misunderstood part of users stories. I have seen detailed stories with no acceptance criteria, criteria that restate the narrative, and criteria that hide new stories, or even contain entire workflows. The last two mistakes are exemplified by the following story (the narrative is on the card on the left, the "acceptance criteria" on the right):

The idea behind acceptance criteria is simple: They allow you to describe the conditions that have to be fulfilled so that the story is done, that it can be exposed to the users and the other stakeholders. This ensures that you gather feedback and/or release features, and it helps the team plan and track their work: The criteria enrich the story and make it more precise and testable. (The criteria all stories have to fulfil such as "online help is available" are not stated in the acceptance criteria but in the Definition of Done.)

As a rule of thumb, I like to work with three to five criteria per story, and I am not worried if my epics don't have acceptance criteria to start with. Ready stories, however, must provide meaningful criteria. You can find sample acceptance criteria in my posts "Epics and Ready Stories" and "Nonfunctional Requirements".

http://www.romanpichler.com/blog/user-stories/writing-good-user-stories/

10 Tips for Writing Good User Stories

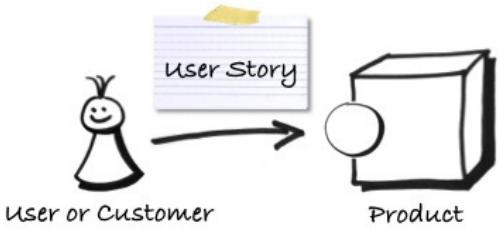
Posted on Thursday 7th October 2010

Summary

User stories are probably the most popular agile technique to capture product functionality: Working with user stories is easy. But writing good stories can be hard. The following ten tips help you create good stories.

1. Focus on the user

As its name suggests, a user story tells a story about a customer or user employing the product. Write therefore stories from the user's perspective and show how a user or customer uses some product functionality.

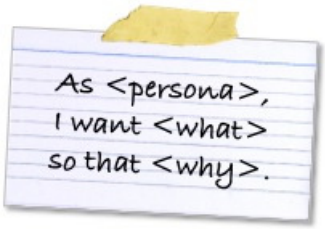


2. Write stories collaboratively

A user story is not a specification, but an communication and collaboration tool. Stories should not be handed off to the development team but be complemented by a conversation: Product owner and team should talk about stories, or even better, write them together: Invite the team to create and detail the stories together. This leverages the creativity and the knowledge of the team and usually results in better user stories.

3. Keep your stories simple and concise

Write your stories so that they are easy to understand. Keep them simple and concise. Avoid confusing and ambiguous terms, and use active voice. Focus on what's important, and leave out non-essential information. The following template puts the user or customer into the story and makes its benefit explicit. But use the template only if it is helpful. Try out different ways to write your stories to understand what works best for you.

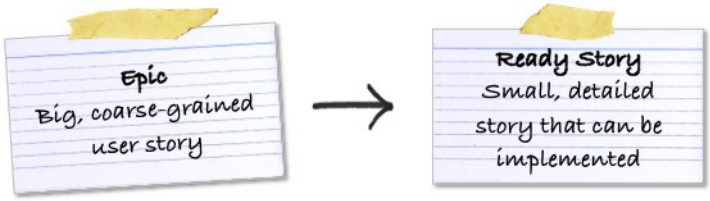


4. Start with epics

Epics are big, coarse-grained user stories. Starting with epics allows you to sketch the product functionality without committing to the details. This is particularly helpful for new products and new features, as it buys you time to learn more about the users and how to best meet their needs. It also reduces the time and effort required to integrate new insights: If you have lots of detailed stories, then it's often tricky to relate the feedback you receive to the right stories. I like to use the personas with their goals to discover the right epics.

5. Decompose your stories until they are ready

Break your epics into smaller, detailed stories until they are ready: clear, feasible, and testable. Everyone should have a shared understanding of the story's meaning; the story should not too big, and there has to be an effective way to determine if the story is done.



6. Add acceptance criteria

As you decompose epics into smaller stories, remember to add acceptance criteria. Acceptance criteria complement the story's narrative: They allow you to describe the conditions that have to be fulfilled so that the story is done. The criteria enrich the story and make it more precise and testable, and they ensures that the story can be demoed or released to the users and the other stakeholders.

7. Group user stories into themes

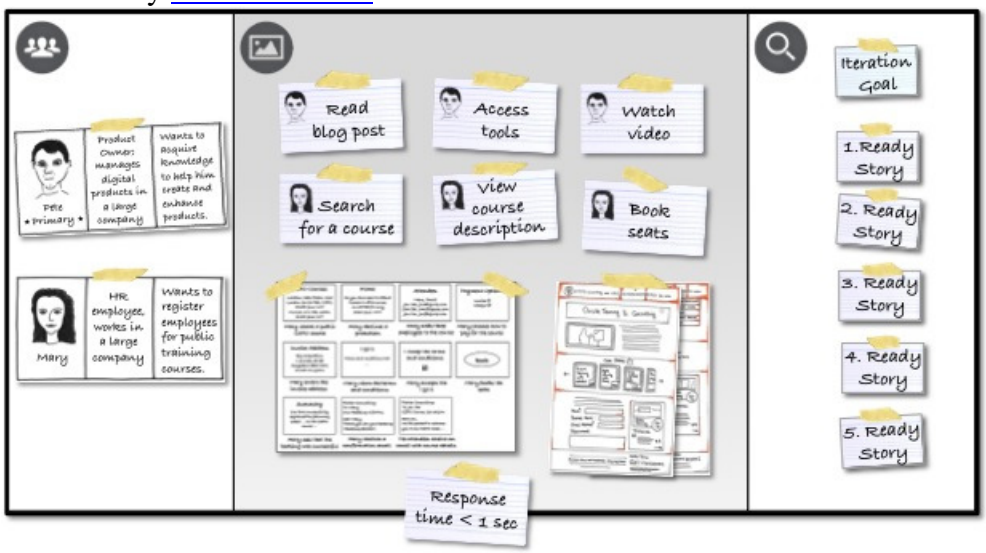
Using themes helps you organise your stories. Each theme is a group of related stories. Sample themes for a mobile phone are email, calendar, voice communication, and organizer, for instance. Themes make it easier to check for completeness and consistency, they structure your product canvas or product backlog, and they facilitate prioritisation.

8. Use paper cards

Paper cards are not only cheap and easy to use. They facilitate collaboration, as everyone can take a card and jot down an idea. Cards can also be easily grouped on the table or wall to check for consistency and completeness. Even if your stories are stored electronically, it is worthwhile to use paper cards when you write new stories.

9. Keep your stories visible and accessible

Stories want to communication information. Don't hide them on a network drive, the corporate intranet jungle, or a licensed tool. Make them visible instead, for instance, by putting them up on the wall. A great tool to discover, visualise, and manage your stories is my Product Canvas.



10. Don't solely rely on user stories

Creating a great user experience (UX) requires more than writing user stories. Also consider the user journeys and interactions, the visual design, and the nonfunctional properties of your product. This results in a holistic description that makes it easier to identify risks and assumptions, and it increases the chances of creating a product with the right user experience.

http://www.romanpichler.com/blog/user-stories/5-common-user-story-mistakes/

5 Common User Story Mistakes

Posted on Wednesday 12th June 2013

Summary

This post discusses five common user story mistakes and helps you overcome them.

User stories are a simple, yet effective way to communicate how a user or customer employs a product. But writing user stories that help a team build great software can be challenging. The following list summarises five common user story mistakes to help you improve your stories.

Story Mania

Some product owners and teams are so fond of user stories that everything is expressed as a story. This either results in some rather odd stories – stories that capture the user interface design, complex user interactions, and technical requirements; or these aspects are simply overlooked.

Like any technique, user story writing has its strengths and limitations. I find stories particularly well suited to capture product functionality, and when applied properly, nonfunctional requirements. But user interface design and complex user interactions are better described by other techniques including design sketches, mock-ups, scenarios, and storyboards. Complement your user stories therefore with other techniques, and don't feel obliged to only use stories.

User Incognito

A user story tells a story about a user interacting with the product. Some stories, however, omit the beneficiary altogether, or they talk about a mysterious user as in "As the user, I want to …" But if it's not clear who the user is, why should the story be developed? How can we be confident that the story will benefit someone?

Make sure that all your stories have a clear user or customer. As I like to work with personas, I use personas in my stories (instead of user roles). This connects each story to the right persona, and it allows me to understand if and to which extend the story addresses the persona's need or goal. To achieve this, I use the following template: As <persona>, I want <something> so that <benefit>.

Disastrous Details

The devil is in the details: Some stories are too big and vague for the team to understand and implement. Others contain too much detail, or even prescribe a solution. Getting the details right seems to be a battle the product owner can only loose.

My recommendation is: Start with big, coarse-grained stories called epics particularly for all new features. Epics allow you to capture an idea without committing to the details. Then break an epic into more detailed user stories. The new user stories replace the epic, and they provide more information about the product's functionality. Pay particular attention to the user stories that are pulled into a sprint. These stories have to be ready: clear, feasible, and testable. (You can learn more about decomposing epics into ready stories in my post "Epics and Ready Stories".)

Progressively refining your stories keeps your Product Canvas or backlog concise, it makes it easier to integrate new insights, and it reduces the effort required to stock your initial canvas or backlog. This is particularly valuable for new products and new features. Make sure you do not prescribe a solution in your stories. Rather focus on the "what", nor the "how". The latter is best captured in an architecture diagram.

Story Handoff

Some product owners diligently write user stories, and give them to the development team in the sprint planning meeting. This handoff is usually suboptimal, as it wastes the team's ideas and knowledge. Stories can hence be inappropriate, difficult to understand, unfeasible and not testable.

User stories, however, are not meant to be standalone documents. They should be complemented by a conversation between the product owner and the team, or even better: written collaboratively. A story wants to capture the essentials, and not specify every detail: The latter would be difficult for new features and too slow and too expensive in an agile / lean context.

User story writing should be a team effort, where product owner and development team create the stories together. Allocate time for a collaborative Product Canvas workshop or backlog grooming meeting, particularly when you develop a new product or new features. Trust me: Better stories and a better product will be your reward.

Criteria Crisis

Acceptance criteria are maybe the most misunderstood part of users stories. I have seen detailed stories with no acceptance criteria, criteria that restate the narrative, and criteria that hide new stories, or even contain entire workflows. The last two mistakes are exemplified by the following story (the narrative is on the card on the left, the "acceptance criteria" on the right):

The idea behind acceptance criteria is simple: They allow you to describe the conditions that have to be fulfilled so that the story is done, that it can be exposed to the users and the other stakeholders. This ensures that you gather feedback and/or release features, and it helps the team plan and track their work: The criteria enrich the story and make it more precise and testable. (The criteria all stories have to fulfil such as "online help is available" are not stated in the acceptance criteria but in the Definition of Done.)

As a rule of thumb, I like to work with three to five criteria per story, and I am not worried if my epics don't have acceptance criteria to start with. Ready stories, however, must provide meaningful criteria. You can find sample acceptance criteria in my posts "Epics and Ready Stories" and "Nonfunctional Requirements".

http://www.romanpichler.com/blog/user-stories/writing-good-user-stories/

10 Tips for Writing Good User Stories

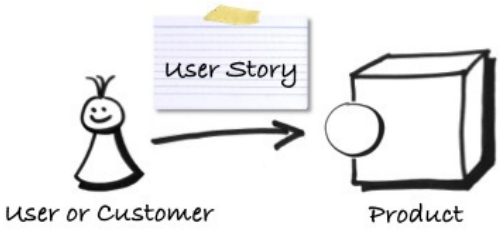
Posted on Thursday 7th October 2010

Summary

User stories are probably the most popular agile technique to capture product functionality: Working with user stories is easy. But writing good stories can be hard. The following ten tips help you create good stories.

1. Focus on the user

As its name suggests, a user story tells a story about a customer or user employing the product. Write therefore stories from the user's perspective and show how a user or customer uses some product functionality.

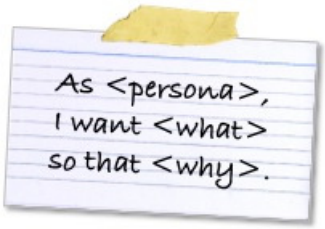


2. Write stories collaboratively

A user story is not a specification, but an communication and collaboration tool. Stories should not be handed off to the development team but be complemented by a conversation: Product owner and team should talk about stories, or even better, write them together: Invite the team to create and detail the stories together. This leverages the creativity and the knowledge of the team and usually results in better user stories.

3. Keep your stories simple and concise

Write your stories so that they are easy to understand. Keep them simple and concise. Avoid confusing and ambiguous terms, and use active voice. Focus on what's important, and leave out non-essential information. The following template puts the user or customer into the story and makes its benefit explicit. But use the template only if it is helpful. Try out different ways to write your stories to understand what works best for you.

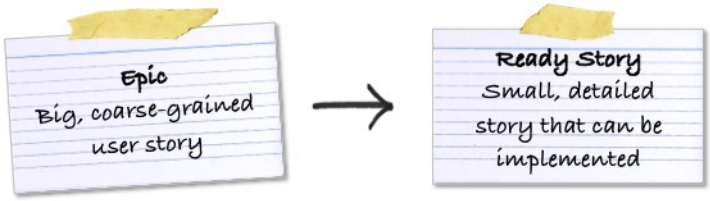


4. Start with epics

Epics are big, coarse-grained user stories. Starting with epics allows you to sketch the product functionality without committing to the details. This is particularly helpful for new products and new features, as it buys you time to learn more about the users and how to best meet their needs. It also reduces the time and effort required to integrate new insights: If you have lots of detailed stories, then it's often tricky to relate the feedback you receive to the right stories. I like to use the personas with their goals to discover the right epics.

5. Decompose your stories until they are ready

Break your epics into smaller, detailed stories until they are ready: clear, feasible, and testable. Everyone should have a shared understanding of the story's meaning; the story should not too big, and there has to be an effective way to determine if the story is done.



6. Add acceptance criteria

As you decompose epics into smaller stories, remember to add acceptance criteria. Acceptance criteria complement the story's narrative: They allow you to describe the conditions that have to be fulfilled so that the story is done. The criteria enrich the story and make it more precise and testable, and they ensures that the story can be demoed or released to the users and the other stakeholders.

7. Group user stories into themes

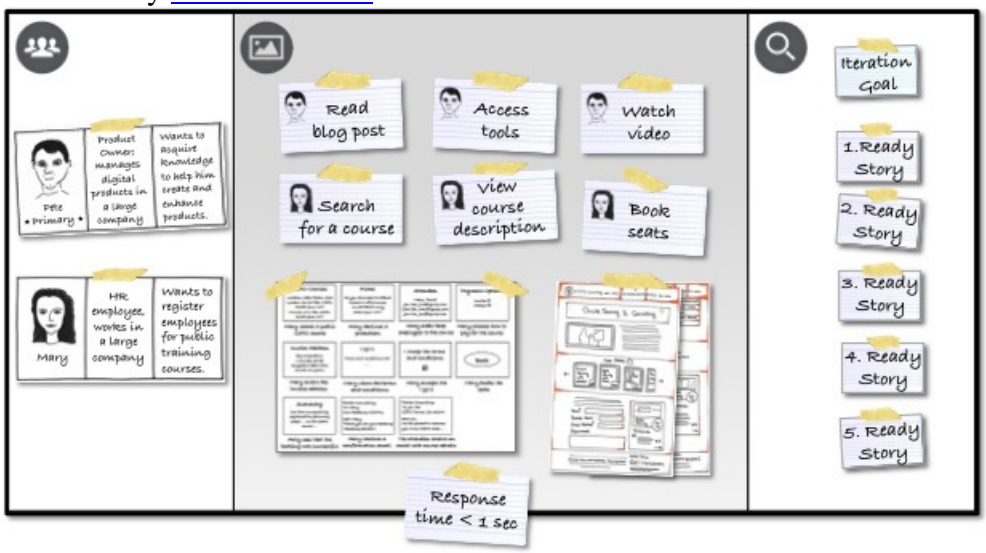
Using themes helps you organise your stories. Each theme is a group of related stories. Sample themes for a mobile phone are email, calendar, voice communication, and organizer, for instance. Themes make it easier to check for completeness and consistency, they structure your product canvas or product backlog, and they facilitate prioritisation.

8. Use paper cards

Paper cards are not only cheap and easy to use. They facilitate collaboration, as everyone can take a card and jot down an idea. Cards can also be easily grouped on the table or wall to check for consistency and completeness. Even if your stories are stored electronically, it is worthwhile to use paper cards when you write new stories.

9. Keep your stories visible and accessible

Stories want to communication information. Don't hide them on a network drive, the corporate intranet jungle, or a licensed tool. Make them visible instead, for instance, by putting them up on the wall. A great tool to discover, visualise, and manage your stories is my Product Canvas.



10. Don't solely rely on user stories

Creating a great user experience (UX) requires more than writing user stories. Also consider the user journeys and interactions, the visual design, and the nonfunctional properties of your product. This results in a holistic description that makes it easier to identify risks and assumptions, and it increases the chances of creating a product with the right user experience.

http://www.romanpichler.com/blog/user-stories/5-common-user-story-mistakes/

5 Common User Story Mistakes

Posted on Wednesday 12th June 2013

Summary

This post discusses five common user story mistakes and helps you overcome them.

User stories are a simple, yet effective way to communicate how a user or customer employs a product. But writing user stories that help a team build great software can be challenging. The following list summarises five common user story mistakes to help you improve your stories.

Story Mania

Some product owners and teams are so fond of user stories that everything is expressed as a story. This either results in some rather odd stories – stories that capture the user interface design, complex user interactions, and technical requirements; or these aspects are simply overlooked.

Like any technique, user story writing has its strengths and limitations. I find stories particularly well suited to capture product functionality, and when applied properly, nonfunctional requirements. But user interface design and complex user interactions are better described by other techniques including design sketches, mock-ups, scenarios, and storyboards. Complement your user stories therefore with other techniques, and don't feel obliged to only use stories.

User Incognito

A user story tells a story about a user interacting with the product. Some stories, however, omit the beneficiary altogether, or they talk about a mysterious user as in "As the user, I want to …" But if it's not clear who the user is, why should the story be developed? How can we be confident that the story will benefit someone?

Make sure that all your stories have a clear user or customer. As I like to work with personas, I use personas in my stories (instead of user roles). This connects each story to the right persona, and it allows me to understand if and to which extend the story addresses the persona's need or goal. To achieve this, I use the following template: As <persona>, I want <something> so that <benefit>.

Disastrous Details

The devil is in the details: Some stories are too big and vague for the team to understand and implement. Others contain too much detail, or even prescribe a solution. Getting the details right seems to be a battle the product owner can only loose.

My recommendation is: Start with big, coarse-grained stories called epics particularly for all new features. Epics allow you to capture an idea without committing to the details. Then break an epic into more detailed user stories. The new user stories replace the epic, and they provide more information about the product's functionality. Pay particular attention to the user stories that are pulled into a sprint. These stories have to be ready: clear, feasible, and testable. (You can learn more about decomposing epics into ready stories in my post "Epics and Ready Stories".)

Progressively refining your stories keeps your Product Canvas or backlog concise, it makes it easier to integrate new insights, and it reduces the effort required to stock your initial canvas or backlog. This is particularly valuable for new products and new features. Make sure you do not prescribe a solution in your stories. Rather focus on the "what", nor the "how". The latter is best captured in an architecture diagram.

Story Handoff

Some product owners diligently write user stories, and give them to the development team in the sprint planning meeting. This handoff is usually suboptimal, as it wastes the team's ideas and knowledge. Stories can hence be inappropriate, difficult to understand, unfeasible and not testable.

User stories, however, are not meant to be standalone documents. They should be complemented by a conversation between the product owner and the team, or even better: written collaboratively. A story wants to capture the essentials, and not specify every detail: The latter would be difficult for new features and too slow and too expensive in an agile / lean context.

User story writing should be a team effort, where product owner and development team create the stories together. Allocate time for a collaborative Product Canvas workshop or backlog grooming meeting, particularly when you develop a new product or new features. Trust me: Better stories and a better product will be your reward.

Criteria Crisis

Acceptance criteria are maybe the most misunderstood part of users stories. I have seen detailed stories with no acceptance criteria, criteria that restate the narrative, and criteria that hide new stories, or even contain entire workflows. The last two mistakes are exemplified by the following story (the narrative is on the card on the left, the "acceptance criteria" on the right):

The idea behind acceptance criteria is simple: They allow you to describe the conditions that have to be fulfilled so that the story is done, that it can be exposed to the users and the other stakeholders. This ensures that you gather feedback and/or release features, and it helps the team plan and track their work: The criteria enrich the story and make it more precise and testable. (The criteria all stories have to fulfil such as "online help is available" are not stated in the acceptance criteria but in the Definition of Done.)

As a rule of thumb, I like to work with three to five criteria per story, and I am not worried if my epics don't have acceptance criteria to start with. Ready stories, however, must provide meaningful criteria. You can find sample acceptance criteria in my posts "Epics and Ready Stories" and "Nonfunctional Requirements".

http://www.romanpichler.com/blog/user-stories/writing-good-user-stories/

10 Tips for Writing Good User Stories

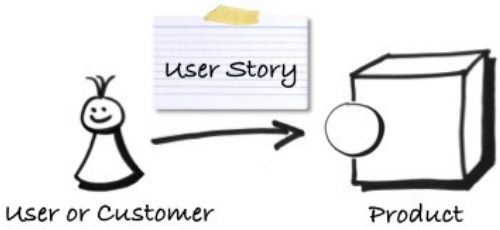
Posted on Thursday 7th October 2010

Summary

User stories are probably the most popular agile technique to capture product functionality: Working with user stories is easy. But writing good stories can be hard. The following ten tips help you create good stories.

1. Focus on the user

As its name suggests, a user story tells a story about a customer or user employing the product. Write therefore stories from the user's perspective and show how a user or customer uses some product functionality.

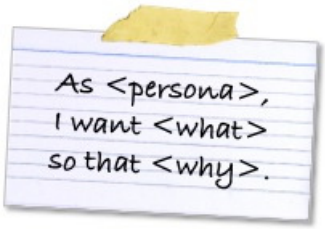


2. Write stories collaboratively

A user story is not a specification, but an communication and collaboration tool. Stories should not be handed off to the development team but be complemented by a conversation: Product owner and team should talk about stories, or even better, write them together: Invite the team to create and detail the stories together. This leverages the creativity and the knowledge of the team and usually results in better user stories.

3. Keep your stories simple and concise

Write your stories so that they are easy to understand. Keep them simple and concise. Avoid confusing and ambiguous terms, and use active voice. Focus on what's important, and leave out non-essential information. The following template puts the user or customer into the story and makes its benefit explicit. But use the template only if it is helpful. Try out different ways to write your stories to understand what works best for you.

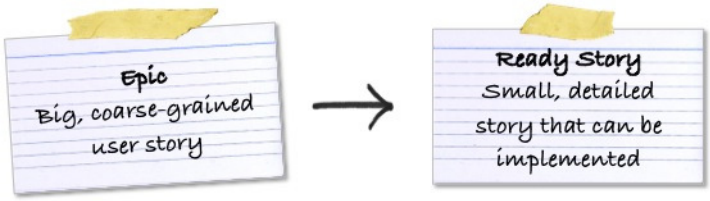


4. Start with epics

Epics are big, coarse-grained user stories. Starting with epics allows you to sketch the product functionality without committing to the details. This is particularly helpful for new products and new features, as it buys you time to learn more about the users and how to best meet their needs. It also reduces the time and effort required to integrate new insights: If you have lots of detailed stories, then it's often tricky to relate the feedback you receive to the right stories. I like to use the personas with their goals to discover the right epics.

5. Decompose your stories until they are ready

Break your epics into smaller, detailed stories until they are ready: clear, feasible, and testable. Everyone should have a shared understanding of the story's meaning; the story should not too big, and there has to be an effective way to determine if the story is done.



6. Add acceptance criteria

As you decompose epics into smaller stories, remember to add acceptance criteria. Acceptance criteria complement the story's narrative: They allow you to describe the conditions that have to be fulfilled so that the story is done. The criteria enrich the story and make it more precise and testable, and they ensures that the story can be demoed or released to the users and the other stakeholders.

7. Group user stories into themes

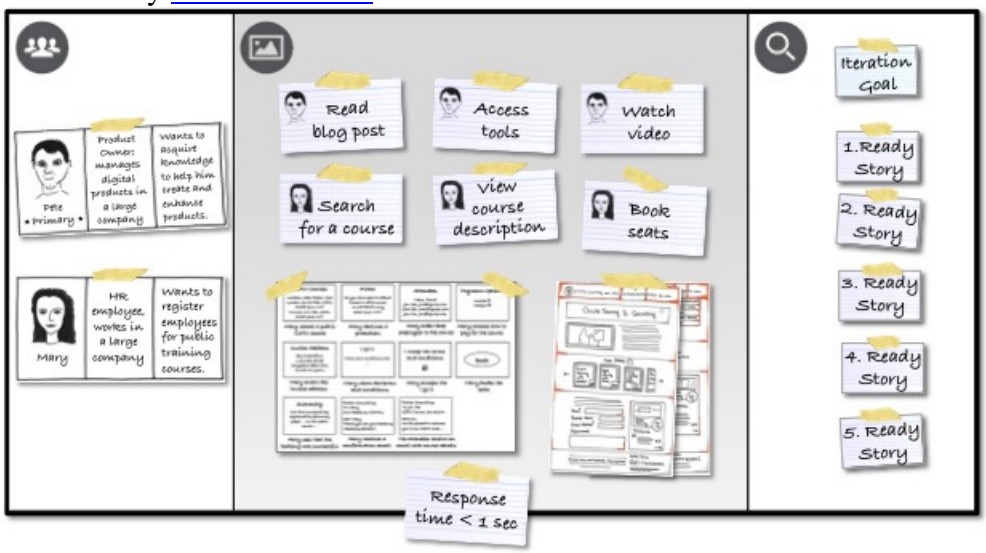
Using themes helps you organise your stories. Each theme is a group of related stories. Sample themes for a mobile phone are email, calendar, voice communication, and organizer, for instance. Themes make it easier to check for completeness and consistency, they structure your product canvas or product backlog, and they facilitate prioritisation.

8. Use paper cards

Paper cards are not only cheap and easy to use. They facilitate collaboration, as everyone can take a card and jot down an idea. Cards can also be easily grouped on the table or wall to check for consistency and completeness. Even if your stories are stored electronically, it is worthwhile to use paper cards when you write new stories.

9. Keep your stories visible and accessible

Stories want to communication information. Don't hide them on a network drive, the corporate intranet jungle, or a licensed tool. Make them visible instead, for instance, by putting them up on the wall. A great tool to discover, visualise, and manage your stories is my Product Canvas.



10. Don't solely rely on user stories

Creating a great user experience (UX) requires more than writing user stories. Also consider the user journeys and interactions, the visual design, and the nonfunctional properties of your product. This results in a holistic description that makes it easier to identify risks and assumptions, and it increases the chances of creating a product with the right user experience.

http://www.romanpichler.com/blog/user-stories/5-common-user-story-mistakes/

5 Common User Story Mistakes

Posted on Wednesday 12th June 2013

Summary

This post discusses five common user story mistakes and helps you overcome them.

User stories are a simple, yet effective way to communicate how a user or customer employs a product. But writing user stories that help a team build great software can be challenging. The following list summarises five common user story mistakes to help you improve your stories.

Story Mania

Some product owners and teams are so fond of user stories that everything is expressed as a story. This either results in some rather odd stories – stories that capture the user interface design, complex user interactions, and technical requirements; or these aspects are simply overlooked.

Like any technique, user story writing has its strengths and limitations. I find stories particularly well suited to capture product functionality, and when applied properly, nonfunctional requirements. But user interface design and complex user interactions are better described by other techniques including design sketches, mock-ups, scenarios, and storyboards. Complement your user stories therefore with other techniques, and don't feel obliged to only use stories.

User Incognito

A user story tells a story about a user interacting with the product. Some stories, however, omit the beneficiary altogether, or they talk about a mysterious user as in "As the user, I want to …" But if it's not clear who the user is, why should the story be developed? How can we be confident that the story will benefit someone?

Make sure that all your stories have a clear user or customer. As I like to work with personas, I use personas in my stories (instead of user roles). This connects each story to the right persona, and it allows me to understand if and to which extend the story addresses the persona's need or goal. To achieve this, I use the following template: As <persona>, I want <something> so that <benefit>.

Disastrous Details

The devil is in the details: Some stories are too big and vague for the team to understand and implement. Others contain too much detail, or even prescribe a solution. Getting the details right seems to be a battle the product owner can only loose.

My recommendation is: Start with big, coarse-grained stories called epics particularly for all new features. Epics allow you to capture an idea without committing to the details. Then break an epic into more detailed user stories. The new user stories replace the epic, and they provide more information about the product's functionality. Pay particular attention to the user stories that are pulled into a sprint. These stories have to be ready: clear, feasible, and testable. (You can learn more about decomposing epics into ready stories in my post "Epics and Ready Stories".)

Progressively refining your stories keeps your Product Canvas or backlog concise, it makes it easier to integrate new insights, and it reduces the effort required to stock your initial canvas or backlog. This is particularly valuable for new products and new features. Make sure you do not prescribe a solution in your stories. Rather focus on the "what", nor the "how". The latter is best captured in an architecture diagram.

Story Handoff

Some product owners diligently write user stories, and give them to the development team in the sprint planning meeting. This handoff is usually suboptimal, as it wastes the team's ideas and knowledge. Stories can hence be inappropriate, difficult to understand, unfeasible and not testable.

User stories, however, are not meant to be standalone documents. They should be complemented by a conversation between the product owner and the team, or even better: written collaboratively. A story wants to capture the essentials, and not specify every detail: The latter would be difficult for new features and too slow and too expensive in an agile / lean context.

User story writing should be a team effort, where product owner and development team create the stories together. Allocate time for a collaborative Product Canvas workshop or backlog grooming meeting, particularly when you develop a new product or new features. Trust me: Better stories and a better product will be your reward.

Criteria Crisis

Acceptance criteria are maybe the most misunderstood part of users stories. I have seen detailed stories with no acceptance criteria, criteria that restate the narrative, and criteria that hide new stories, or even contain entire workflows. The last two mistakes are exemplified by the following story (the narrative is on the card on the left, the "acceptance criteria" on the right):

The idea behind acceptance criteria is simple: They allow you to describe the conditions that have to be fulfilled so that the story is done, that it can be exposed to the users and the other stakeholders. This ensures that you gather feedback and/or release features, and it helps the team plan and track their work: The criteria enrich the story and make it more precise and testable. (The criteria all stories have to fulfil such as "online help is available" are not stated in the acceptance criteria but in the Definition of Done.)

As a rule of thumb, I like to work with three to five criteria per story, and I am not worried if my epics don't have acceptance criteria to start with. Ready stories, however, must provide meaningful criteria. You can find sample acceptance criteria in my posts "Epics and Ready Stories" and "Nonfunctional Requirements".

http://www.romanpichler.com/blog/user-stories/writing-good-user-stories/

10 Tips for Writing Good User Stories

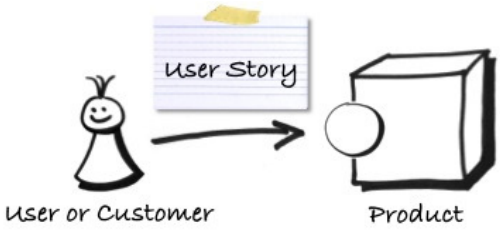
Posted on Thursday 7th October 2010

Summary

User stories are probably the most popular agile technique to capture product functionality: Working with user stories is easy. But writing good stories can be hard. The following ten tips help you create good stories.

1. Focus on the user

As its name suggests, a user story tells a story about a customer or user employing the product. Write therefore stories from the user's perspective and show how a user or customer uses some product functionality.

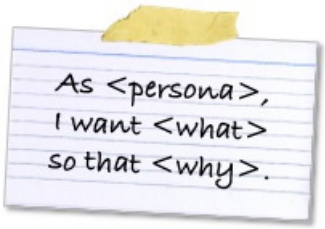


2. Write stories collaboratively

A user story is not a specification, but an communication and collaboration tool. Stories should not be handed off to the development team but be complemented by a conversation: Product owner and team should talk about stories, or even better, write them together: Invite the team to create and detail the stories together. This leverages the creativity and the knowledge of the team and usually results in better user stories.

3. Keep your stories simple and concise

Write your stories so that they are easy to understand. Keep them simple and concise. Avoid confusing and ambiguous terms, and use active voice. Focus on what's important, and leave out non-essential information. The following template puts the user or customer into the story and makes its benefit explicit. But use the template only if it is helpful. Try out different ways to write your stories to understand what works best for you.

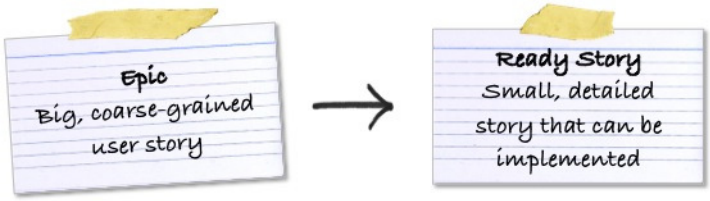


4. Start with epics

Epics are big, coarse-grained user stories. Starting with epics allows you to sketch the product functionality without committing to the details. This is particularly helpful for new products and new features, as it buys you time to learn more about the users and how to best meet their needs. It also reduces the time and effort required to integrate new insights: If you have lots of detailed stories, then it's often tricky to relate the feedback you receive to the right stories. I like to use the personas with their goals to discover the right epics.

5. Decompose your stories until they are ready

Break your epics into smaller, detailed stories until they are ready: clear, feasible, and testable. Everyone should have a shared understanding of the story's meaning; the story should not too big, and there has to be an effective way to determine if the story is done.



6. Add acceptance criteria

As you decompose epics into smaller stories, remember to add acceptance criteria. Acceptance criteria complement the story's narrative: They allow you to describe the conditions that have to be fulfilled so that the story is done. The criteria enrich the story and make it more precise and testable, and they ensures that the story can be demoed or released to the users and the other stakeholders.

7. Group user stories into themes

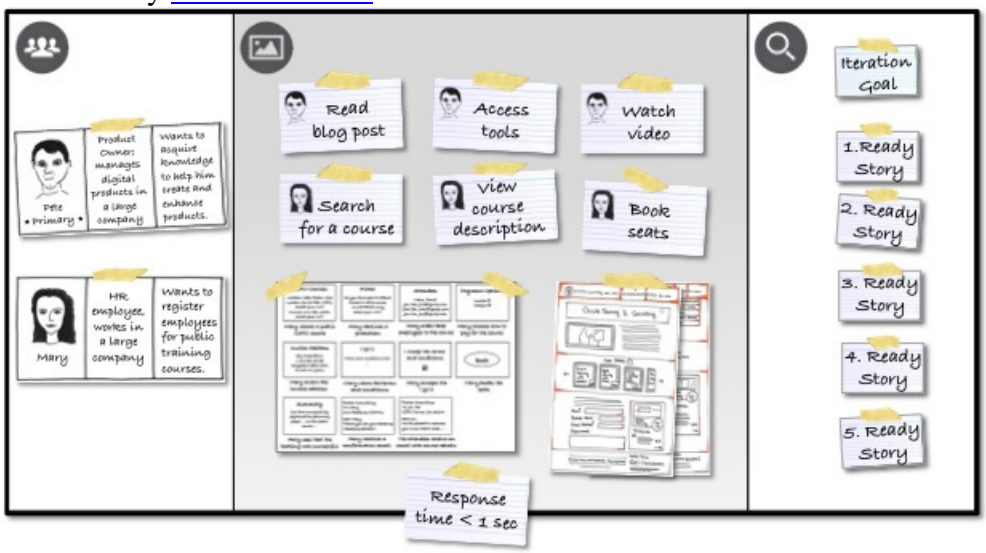
Using themes helps you organise your stories. Each theme is a group of related stories. Sample themes for a mobile phone are email, calendar, voice communication, and organizer, for instance. Themes make it easier to check for completeness and consistency, they structure your product canvas or product backlog, and they facilitate prioritisation.

8. Use paper cards

Paper cards are not only cheap and easy to use. They facilitate collaboration, as everyone can take a card and jot down an idea. Cards can also be easily grouped on the table or wall to check for consistency and completeness. Even if your stories are stored electronically, it is worthwhile to use paper cards when you write new stories.

9. Keep your stories visible and accessible

Stories want to communication information. Don't hide them on a network drive, the corporate intranet jungle, or a licensed tool. Make them visible instead, for instance, by putting them up on the wall. A great tool to discover, visualise, and manage your stories is my Product Canvas.



10. Don't solely rely on user stories

Creating a great user experience (UX) requires more than writing user stories. Also consider the user journeys and interactions, the visual design, and the nonfunctional properties of your product. This results in a holistic description that makes it easier to identify risks and assumptions, and it increases the chances of creating a product with the right user experience.

http://www.romanpichler.com/blog/user-stories/5-common-user-story-mistakes/

5 Common User Story Mistakes

Posted on Wednesday 12th June 2013

Summary

This post discusses five common user story mistakes and helps you overcome them.

User stories are a simple, yet effective way to communicate how a user or customer employs a product. But writing user stories that help a team build great software can be challenging. The following list summarises five common user story mistakes to help you improve your stories.

Story Mania

Some product owners and teams are so fond of user stories that everything is expressed as a story. This either results in some rather odd stories – stories that capture the user interface design, complex user interactions, and technical requirements; or these aspects are simply overlooked.

Like any technique, user story writing has its strengths and limitations. I find stories particularly well suited to capture product functionality, and when applied properly, nonfunctional requirements. But user interface design and complex user interactions are better described by other techniques including design sketches, mock-ups, scenarios, and storyboards. Complement your user stories therefore with other techniques, and don't feel obliged to only use stories.

User Incognito

A user story tells a story about a user interacting with the product. Some stories, however, omit the beneficiary altogether, or they talk about a mysterious user as in "As the user, I want to …" But if it's not clear who the user is, why should the story be developed? How can we be confident that the story will benefit someone?

Make sure that all your stories have a clear user or customer. As I like to work with personas, I use personas in my stories (instead of user roles). This connects each story to the right persona, and it allows me to understand if and to which extend the story addresses the persona's need or goal. To achieve this, I use the following template: As <persona>, I want <something> so that <benefit>.

Disastrous Details

The devil is in the details: Some stories are too big and vague for the team to understand and implement. Others contain too much detail, or even prescribe a solution. Getting the details right seems to be a battle the product owner can only loose.

My recommendation is: Start with big, coarse-grained stories called epics particularly for all new features. Epics allow you to capture an idea without committing to the details. Then break an epic into more detailed user stories. The new user stories replace the epic, and they provide more information about the product's functionality. Pay particular attention to the user stories that are pulled into a sprint. These stories have to be ready: clear, feasible, and testable. (You can learn more about decomposing epics into ready stories in my post "Epics and Ready Stories".)

Progressively refining your stories keeps your Product Canvas or backlog concise, it makes it easier to integrate new insights, and it reduces the effort required to stock your initial canvas or backlog. This is particularly valuable for new products and new features. Make sure you do not prescribe a solution in your stories. Rather focus on the "what", nor the "how". The latter is best captured in an architecture diagram.

Story Handoff

Some product owners diligently write user stories, and give them to the development team in the sprint planning meeting. This handoff is usually suboptimal, as it wastes the team's ideas and knowledge. Stories can hence be inappropriate, difficult to understand, unfeasible and not testable.

User stories, however, are not meant to be standalone documents. They should be complemented by a conversation between the product owner and the team, or even better: written collaboratively. A story wants to capture the essentials, and not specify every detail: The latter would be difficult for new features and too slow and too expensive in an agile / lean context.

User story writing should be a team effort, where product owner and development team create the stories together. Allocate time for a collaborative Product Canvas workshop or backlog grooming meeting, particularly when you develop a new product or new features. Trust me: Better stories and a better product will be your reward.

Criteria Crisis

Acceptance criteria are maybe the most misunderstood part of users stories. I have seen detailed stories with no acceptance criteria, criteria that restate the narrative, and criteria that hide new stories, or even contain entire workflows. The last two mistakes are exemplified by the following story (the narrative is on the card on the left, the "acceptance criteria" on the right):

The idea behind acceptance criteria is simple: They allow you to describe the conditions that have to be fulfilled so that the story is done, that it can be exposed to the users and the other stakeholders. This ensures that you gather feedback and/or release features, and it helps the team plan and track their work: The criteria enrich the story and make it more precise and testable. (The criteria all stories have to fulfil such as "online help is available" are not stated in the acceptance criteria but in the Definition of Done.)

As a rule of thumb, I like to work with three to five criteria per story, and I am not worried if my epics don't have acceptance criteria to start with. Ready stories, however, must provide meaningful criteria. You can find sample acceptance criteria in my posts "Epics and Ready Stories" and "Nonfunctional Requirements".

http://www.romanpichler.com/blog/user-stories/writing-good-user-stories/

10 Tips for Writing Good User Stories

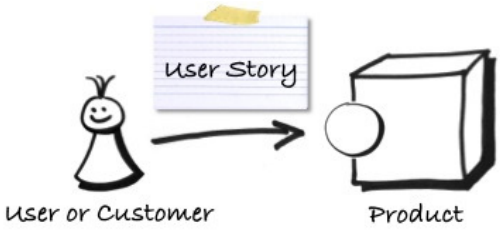
Posted on Thursday 7th October 2010

Summary

User stories are probably the most popular agile technique to capture product functionality: Working with user stories is easy. But writing good stories can be hard. The following ten tips help you create good stories.

1. Focus on the user

As its name suggests, a user story tells a story about a customer or user employing the product. Write therefore stories from the user's perspective and show how a user or customer uses some product functionality.

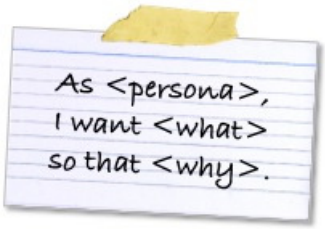


2. Write stories collaboratively

A user story is not a specification, but an communication and collaboration tool. Stories should not be handed off to the development team but be complemented by a conversation: Product owner and team should talk about stories, or even better, write them together: Invite the team to create and detail the stories together. This leverages the creativity and the knowledge of the team and usually results in better user stories.

3. Keep your stories simple and concise

Write your stories so that they are easy to understand. Keep them simple and concise. Avoid confusing and ambiguous terms, and use active voice. Focus on what's important, and leave out non-essential information. The following template puts the user or customer into the story and makes its benefit explicit. But use the template only if it is helpful. Try out different ways to write your stories to understand what works best for you.

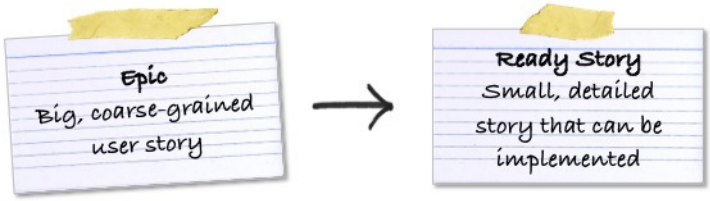


4. Start with epics

Epics are big, coarse-grained user stories. Starting with epics allows you to sketch the product functionality without committing to the details. This is particularly helpful for new products and new features, as it buys you time to learn more about the users and how to best meet their needs. It also reduces the time and effort required to integrate new insights: If you have lots of detailed stories, then it's often tricky to relate the feedback you receive to the right stories. I like to use the personas with their goals to discover the right epics.

5. Decompose your stories until they are ready

Break your epics into smaller, detailed stories until they are ready: clear, feasible, and testable. Everyone should have a shared understanding of the story's meaning; the story should not too big, and there has to be an effective way to determine if the story is done.



6. Add acceptance criteria

As you decompose epics into smaller stories, remember to add acceptance criteria. Acceptance criteria complement the story's narrative: They allow you to describe the conditions that have to be fulfilled so that the story is done. The criteria enrich the story and make it more precise and testable, and they ensures that the story can be demoed or released to the users and the other stakeholders.

7. Group user stories into themes

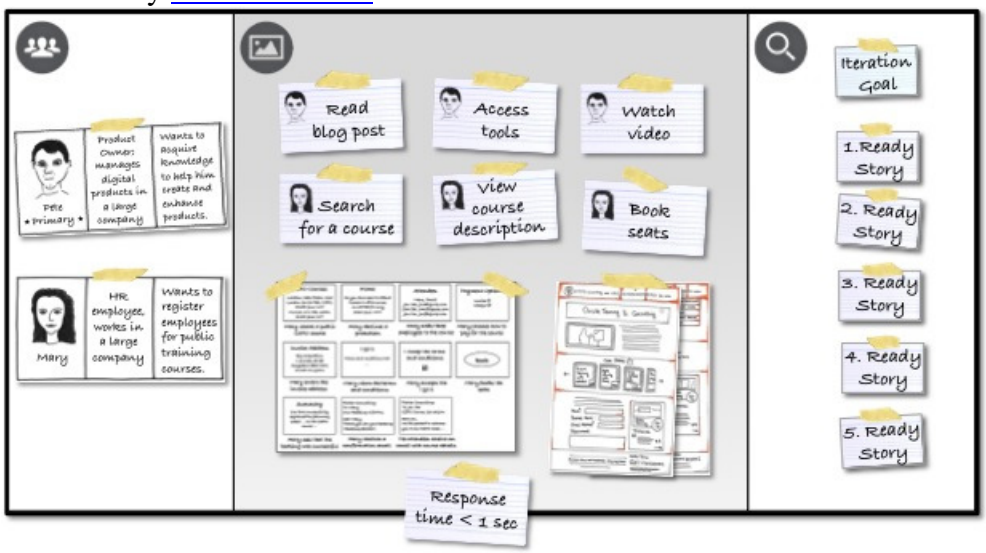
Using themes helps you organise your stories. Each theme is a group of related stories. Sample themes for a mobile phone are email, calendar, voice communication, and organizer, for instance. Themes make it easier to check for completeness and consistency, they structure your product canvas or product backlog, and they facilitate prioritisation.

8. Use paper cards

Paper cards are not only cheap and easy to use. They facilitate collaboration, as everyone can take a card and jot down an idea. Cards can also be easily grouped on the table or wall to check for consistency and completeness. Even if your stories are stored electronically, it is worthwhile to use paper cards when you write new stories.

9. Keep your stories visible and accessible

Stories want to communication information. Don't hide them on a network drive, the corporate intranet jungle, or a licensed tool. Make them visible instead, for instance, by putting them up on the wall. A great tool to discover, visualise, and manage your stories is my Product Canvas.



10. Don't solely rely on user stories

Creating a great user experience (UX) requires more than writing user stories. Also consider the user journeys and interactions, the visual design, and the nonfunctional properties of your product. This results in a holistic description that makes it easier to identify risks and assumptions, and it increases the chances of creating a product with the right user experience.

http://www.romanpichler.com/blog/user-stories/5-common-user-story-mistakes/

5 Common User Story Mistakes

Posted on Wednesday 12th June 2013

Summary

This post discusses five common user story mistakes and helps you overcome them.

User stories are a simple, yet effective way to communicate how a user or customer employs a product. But writing user stories that help a team build great software can be challenging. The following list summarises five common user story mistakes to help you improve your stories.

Story Mania

Some product owners and teams are so fond of user stories that everything is expressed as a story. This either results in some rather odd stories – stories that capture the user interface design, complex user interactions, and technical requirements; or these aspects are simply overlooked.

Like any technique, user story writing has its strengths and limitations. I find stories particularly well suited to capture product functionality, and when applied properly, nonfunctional requirements. But user interface design and complex user interactions are better described by other techniques including design sketches, mock-ups, scenarios, and storyboards. Complement your user stories therefore with other techniques, and don't feel obliged to only use stories.

User Incognito

A user story tells a story about a user interacting with the product. Some stories, however, omit the beneficiary altogether, or they talk about a mysterious user as in "As the user, I want to …" But if it's not clear who the user is, why should the story be developed? How can we be confident that the story will benefit someone?

Make sure that all your stories have a clear user or customer. As I like to work with personas, I use personas in my stories (instead of user roles). This connects each story to the right persona, and it allows me to understand if and to which extend the story addresses the persona's need or goal. To achieve this, I use the following template: As <persona>, I want <something> so that <benefit>.

Disastrous Details

The devil is in the details: Some stories are too big and vague for the team to understand and implement. Others contain too much detail, or even prescribe a solution. Getting the details right seems to be a battle the product owner can only loose.

My recommendation is: Start with big, coarse-grained stories called epics particularly for all new features. Epics allow you to capture an idea without committing to the details. Then break an epic into more detailed user stories. The new user stories replace the epic, and they provide more information about the product's functionality. Pay particular attention to the user stories that are pulled into a sprint. These stories have to be ready: clear, feasible, and testable. (You can learn more about decomposing epics into ready stories in my post "Epics and Ready Stories".)

Progressively refining your stories keeps your Product Canvas or backlog concise, it makes it easier to integrate new insights, and it reduces the effort required to stock your initial canvas or backlog. This is particularly valuable for new products and new features. Make sure you do not prescribe a solution in your stories. Rather focus on the "what", nor the "how". The latter is best captured in an architecture diagram.

Story Handoff

Some product owners diligently write user stories, and give them to the development team in the sprint planning meeting. This handoff is usually suboptimal, as it wastes the team's ideas and knowledge. Stories can hence be inappropriate, difficult to understand, unfeasible and not testable.

User stories, however, are not meant to be standalone documents. They should be complemented by a conversation between the product owner and the team, or even better: written collaboratively. A story wants to capture the essentials, and not specify every detail: The latter would be difficult for new features and too slow and too expensive in an agile / lean context.

User story writing should be a team effort, where product owner and development team create the stories together. Allocate time for a collaborative Product Canvas workshop or backlog grooming meeting, particularly when you develop a new product or new features. Trust me: Better stories and a better product will be your reward.

Criteria Crisis

Acceptance criteria are maybe the most misunderstood part of users stories. I have seen detailed stories with no acceptance criteria, criteria that restate the narrative, and criteria that hide new stories, or even contain entire workflows. The last two mistakes are exemplified by the following story (the narrative is on the card on the left, the "acceptance criteria" on the right):

The idea behind acceptance criteria is simple: They allow you to describe the conditions that have to be fulfilled so that the story is done, that it can be exposed to the users and the other stakeholders. This ensures that you gather feedback and/or release features, and it helps the team plan and track their work: The criteria enrich the story and make it more precise and testable. (The criteria all stories have to fulfil such as "online help is available" are not stated in the acceptance criteria but in the Definition of Done.)

As a rule of thumb, I like to work with three to five criteria per story, and I am not worried if my epics don't have acceptance criteria to start with. Ready stories, however, must provide meaningful criteria. You can find sample acceptance criteria in my posts "Epics and Ready Stories" and "Nonfunctional Requirements".

http://www.romanpichler.com/blog/user-stories/writing-good-user-stories/

10 Tips for Writing Good User Stories

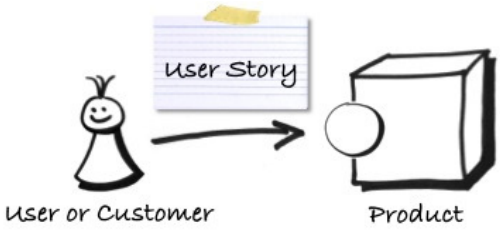
Posted on Thursday 7th October 2010

Summary

User stories are probably the most popular agile technique to capture product functionality: Working with user stories is easy. But writing good stories can be hard. The following ten tips help you create good stories.

1. Focus on the user

As its name suggests, a user story tells a story about a customer or user employing the product. Write therefore stories from the user's perspective and show how a user or customer uses some product functionality.

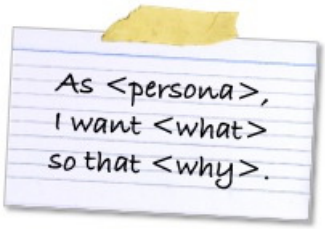


2. Write stories collaboratively

A user story is not a specification, but an communication and collaboration tool. Stories should not be handed off to the development team but be complemented by a conversation: Product owner and team should talk about stories, or even better, write them together: Invite the team to create and detail the stories together. This leverages the creativity and the knowledge of the team and usually results in better user stories.

3. Keep your stories simple and concise

Write your stories so that they are easy to understand. Keep them simple and concise. Avoid confusing and ambiguous terms, and use active voice. Focus on what's important, and leave out non-essential information. The following template puts the user or customer into the story and makes its benefit explicit. But use the template only if it is helpful. Try out different ways to write your stories to understand what works best for you.

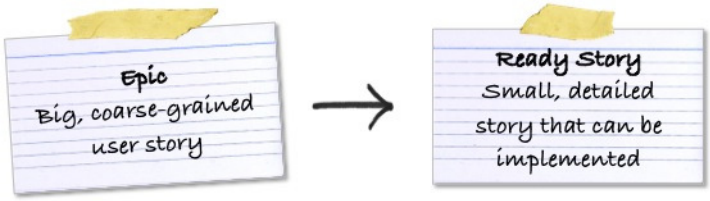


4. Start with epics

Epics are big, coarse-grained user stories. Starting with epics allows you to sketch the product functionality without committing to the details. This is particularly helpful for new products and new features, as it buys you time to learn more about the users and how to best meet their needs. It also reduces the time and effort required to integrate new insights: If you have lots of detailed stories, then it's often tricky to relate the feedback you receive to the right stories. I like to use the personas with their goals to discover the right epics.

5. Decompose your stories until they are ready

Break your epics into smaller, detailed stories until they are ready: clear, feasible, and testable. Everyone should have a shared understanding of the story's meaning; the story should not too big, and there has to be an effective way to determine if the story is done.



6. Add acceptance criteria

As you decompose epics into smaller stories, remember to add acceptance criteria. Acceptance criteria complement the story's narrative: They allow you to describe the conditions that have to be fulfilled so that the story is done. The criteria enrich the story and make it more precise and testable, and they ensures that the story can be demoed or released to the users and the other stakeholders.

7. Group user stories into themes

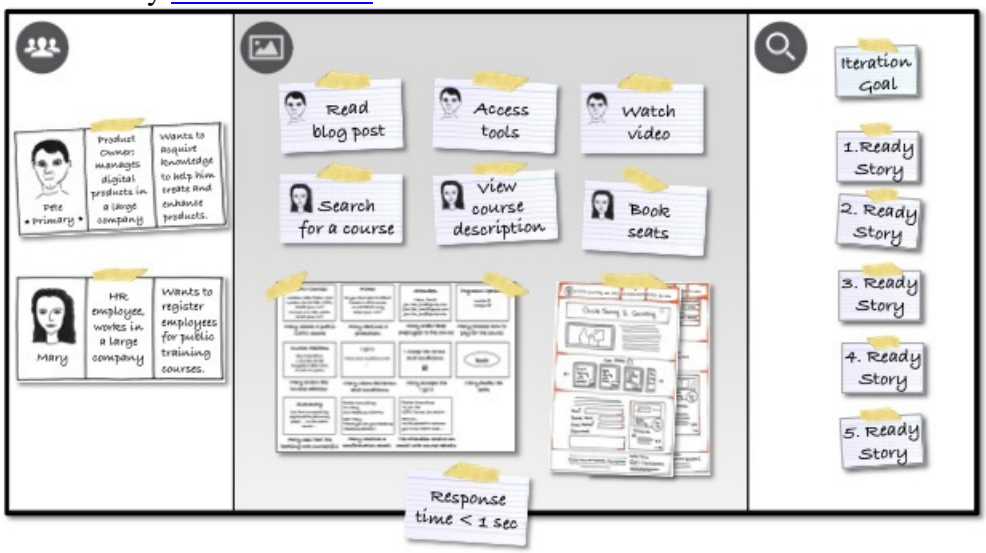
Using themes helps you organise your stories. Each theme is a group of related stories. Sample themes for a mobile phone are email, calendar, voice communication, and organizer, for instance. Themes make it easier to check for completeness and consistency, they structure your product canvas or product backlog, and they facilitate prioritisation.

8. Use paper cards

Paper cards are not only cheap and easy to use. They facilitate collaboration, as everyone can take a card and jot down an idea. Cards can also be easily grouped on the table or wall to check for consistency and completeness. Even if your stories are stored electronically, it is worthwhile to use paper cards when you write new stories.

9. Keep your stories visible and accessible

Stories want to communication information. Don't hide them on a network drive, the corporate intranet jungle, or a licensed tool. Make them visible instead, for instance, by putting them up on the wall. A great tool to discover, visualise, and manage your stories is my Product Canvas.



10. Don't solely rely on user stories

Creating a great user experience (UX) requires more than writing user stories. Also consider the user journeys and interactions, the visual design, and the nonfunctional properties of your product. This results in a holistic description that makes it easier to identify risks and assumptions, and it increases the chances of creating a product with the right user experience.

http://www.romanpichler.com/blog/user-stories/5-common-user-story-mistakes/

5 Common User Story Mistakes

Posted on Wednesday 12th June 2013

Summary

This post discusses five common user story mistakes and helps you overcome them.

User stories are a simple, yet effective way to communicate how a user or customer employs a product. But writing user stories that help a team build great software can be challenging. The following list summarises five common user story mistakes to help you improve your stories.

Story Mania

Some product owners and teams are so fond of user stories that everything is expressed as a story. This either results in some rather odd stories – stories that capture the user interface design, complex user interactions, and technical requirements; or these aspects are simply overlooked.

Like any technique, user story writing has its strengths and limitations. I find stories particularly well suited to capture product functionality, and when applied properly, nonfunctional requirements. But user interface design and complex user interactions are better described by other techniques including design sketches, mock-ups, scenarios, and storyboards. Complement your user stories therefore with other techniques, and don't feel obliged to only use stories.

User Incognito

A user story tells a story about a user interacting with the product. Some stories, however, omit the beneficiary altogether, or they talk about a mysterious user as in "As the user, I want to …" But if it's not clear who the user is, why should the story be developed? How can we be confident that the story will benefit someone?

Make sure that all your stories have a clear user or customer. As I like to work with personas, I use personas in my stories (instead of user roles). This connects each story to the right persona, and it allows me to understand if and to which extend the story addresses the persona's need or goal. To achieve this, I use the following template: As <persona>, I want <something> so that <benefit>.

Disastrous Details

The devil is in the details: Some stories are too big and vague for the team to understand and implement. Others contain too much detail, or even prescribe a solution. Getting the details right seems to be a battle the product owner can only loose.

My recommendation is: Start with big, coarse-grained stories called epics particularly for all new features. Epics allow you to capture an idea without committing to the details. Then break an epic into more detailed user stories. The new user stories replace the epic, and they provide more information about the product's functionality. Pay particular attention to the user stories that are pulled into a sprint. These stories have to be ready: clear, feasible, and testable. (You can learn more about decomposing epics into ready stories in my post "Epics and Ready Stories".)

Progressively refining your stories keeps your Product Canvas or backlog concise, it makes it easier to integrate new insights, and it reduces the effort required to stock your initial canvas or backlog. This is particularly valuable for new products and new features. Make sure you do not prescribe a solution in your stories. Rather focus on the "what", nor the "how". The latter is best captured in an architecture diagram.

Story Handoff

Some product owners diligently write user stories, and give them to the development team in the sprint planning meeting. This handoff is usually suboptimal, as it wastes the team's ideas and knowledge. Stories can hence be inappropriate, difficult to understand, unfeasible and not testable.

User stories, however, are not meant to be standalone documents. They should be complemented by a conversation between the product owner and the team, or even better: written collaboratively. A story wants to capture the essentials, and not specify every detail: The latter would be difficult for new features and too slow and too expensive in an agile / lean context.

User story writing should be a team effort, where product owner and development team create the stories together. Allocate time for a collaborative Product Canvas workshop or backlog grooming meeting, particularly when you develop a new product or new features. Trust me: Better stories and a better product will be your reward.

Criteria Crisis

Acceptance criteria are maybe the most misunderstood part of users stories. I have seen detailed stories with no acceptance criteria, criteria that restate the narrative, and criteria that hide new stories, or even contain entire workflows. The last two mistakes are exemplified by the following story (the narrative is on the card on the left, the "acceptance criteria" on the right):

The idea behind acceptance criteria is simple: They allow you to describe the conditions that have to be fulfilled so that the story is done, that it can be exposed to the users and the other stakeholders. This ensures that you gather feedback and/or release features, and it helps the team plan and track their work: The criteria enrich the story and make it more precise and testable. (The criteria all stories have to fulfil such as "online help is available" are not stated in the acceptance criteria but in the Definition of Done.)

As a rule of thumb, I like to work with three to five criteria per story, and I am not worried if my epics don't have acceptance criteria to start with. Ready stories, however, must provide meaningful criteria. You can find sample acceptance criteria in my posts "Epics and Ready Stories" and "Nonfunctional Requirements".

http://www.romanpichler.com/blog/user-stories/writing-good-user-stories/

10 Tips for Writing Good User Stories

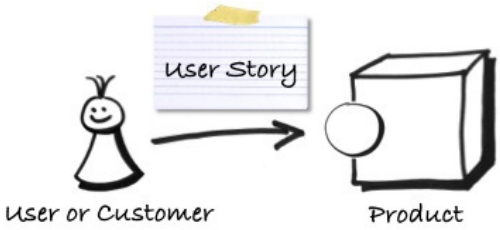
Posted on Thursday 7th October 2010

Summary

User stories are probably the most popular agile technique to capture product functionality: Working with user stories is easy. But writing good stories can be hard. The following ten tips help you create good stories.

1. Focus on the user

As its name suggests, a user story tells a story about a customer or user employing the product. Write therefore stories from the user's perspective and show how a user or customer uses some product functionality.

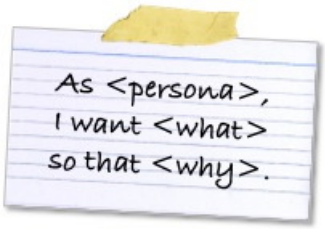


2. Write stories collaboratively

A user story is not a specification, but an communication and collaboration tool. Stories should not be handed off to the development team but be complemented by a conversation: Product owner and team should talk about stories, or even better, write them together: Invite the team to create and detail the stories together. This leverages the creativity and the knowledge of the team and usually results in better user stories.

3. Keep your stories simple and concise

Write your stories so that they are easy to understand. Keep them simple and concise. Avoid confusing and ambiguous terms, and use active voice. Focus on what's important, and leave out non-essential information. The following template puts the user or customer into the story and makes its benefit explicit. But use the template only if it is helpful. Try out different ways to write your stories to understand what works best for you.

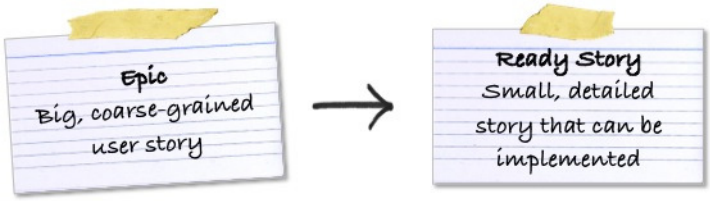


4. Start with epics

Epics are big, coarse-grained user stories. Starting with epics allows you to sketch the product functionality without committing to the details. This is particularly helpful for new products and new features, as it buys you time to learn more about the users and how to best meet their needs. It also reduces the time and effort required to integrate new insights: If you have lots of detailed stories, then it's often tricky to relate the feedback you receive to the right stories. I like to use the personas with their goals to discover the right epics.

5. Decompose your stories until they are ready

Break your epics into smaller, detailed stories until they are ready: clear, feasible, and testable. Everyone should have a shared understanding of the story's meaning; the story should not too big, and there has to be an effective way to determine if the story is done.



6. Add acceptance criteria

As you decompose epics into smaller stories, remember to add acceptance criteria. Acceptance criteria complement the story's narrative: They allow you to describe the conditions that have to be fulfilled so that the story is done. The criteria enrich the story and make it more precise and testable, and they ensures that the story can be demoed or released to the users and the other stakeholders.

7. Group user stories into themes

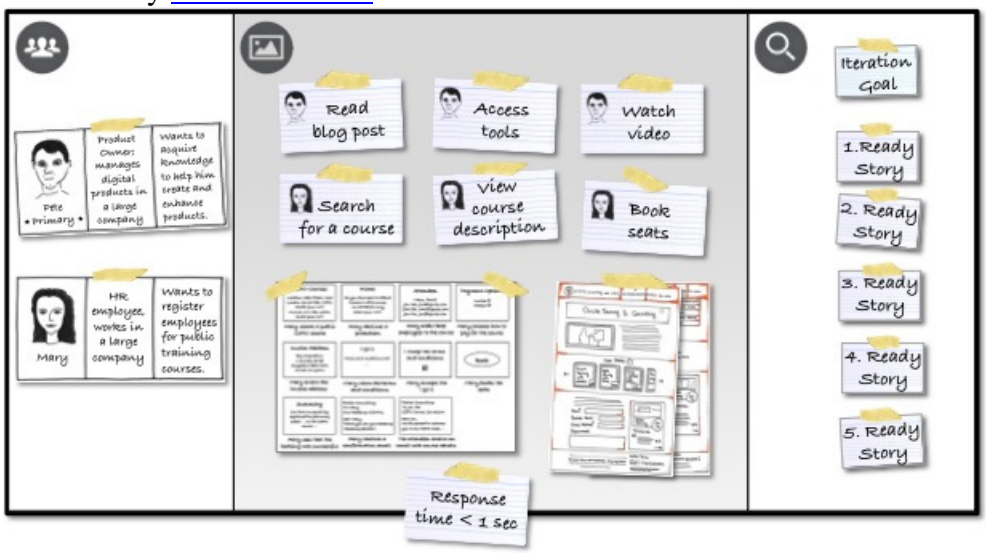
Using themes helps you organise your stories. Each theme is a group of related stories. Sample themes for a mobile phone are email, calendar, voice communication, and organizer, for instance. Themes make it easier to check for completeness and consistency, they structure your product canvas or product backlog, and they facilitate prioritisation.

8. Use paper cards

Paper cards are not only cheap and easy to use. They facilitate collaboration, as everyone can take a card and jot down an idea. Cards can also be easily grouped on the table or wall to check for consistency and completeness. Even if your stories are stored electronically, it is worthwhile to use paper cards when you write new stories.

9. Keep your stories visible and accessible

Stories want to communication information. Don't hide them on a network drive, the corporate intranet jungle, or a licensed tool. Make them visible instead, for instance, by putting them up on the wall. A great tool to discover, visualise, and manage your stories is my Product Canvas.



10. Don't solely rely on user stories

Creating a great user experience (UX) requires more than writing user stories. Also consider the user journeys and interactions, the visual design, and the nonfunctional properties of your product. This results in a holistic description that makes it easier to identify risks and assumptions, and it increases the chances of creating a product with the right user experience.

http://www.romanpichler.com/blog/user-stories/5-common-user-story-mistakes/

5 Common User Story Mistakes

Posted on Wednesday 12th June 2013

Summary

This post discusses five common user story mistakes and helps you overcome them.

User stories are a simple, yet effective way to communicate how a user or customer employs a product. But writing user stories that help a team build great software can be challenging. The following list summarises five common user story mistakes to help you improve your stories.

Story Mania

Some product owners and teams are so fond of user stories that everything is expressed as a story. This either results in some rather odd stories – stories that capture the user interface design, complex user interactions, and technical requirements; or these aspects are simply overlooked.

Like any technique, user story writing has its strengths and limitations. I find stories particularly well suited to capture product functionality, and when applied properly, nonfunctional requirements. But user interface design and complex user interactions are better described by other techniques including design sketches, mock-ups, scenarios, and storyboards. Complement your user stories therefore with other techniques, and don't feel obliged to only use stories.

User Incognito

A user story tells a story about a user interacting with the product. Some stories, however, omit the beneficiary altogether, or they talk about a mysterious user as in "As the user, I want to …" But if it's not clear who the user is, why should the story be developed? How can we be confident that the story will benefit someone?

Make sure that all your stories have a clear user or customer. As I like to work with personas, I use personas in my stories (instead of user roles). This connects each story to the right persona, and it allows me to understand if and to which extend the story addresses the persona's need or goal. To achieve this, I use the following template: As <persona>, I want <something> so that <benefit>.

Disastrous Details

The devil is in the details: Some stories are too big and vague for the team to understand and implement. Others contain too much detail, or even prescribe a solution. Getting the details right seems to be a battle the product owner can only loose.

My recommendation is: Start with big, coarse-grained stories called epics particularly for all new features. Epics allow you to capture an idea without committing to the details. Then break an epic into more detailed user stories. The new user stories replace the epic, and they provide more information about the product's functionality. Pay particular attention to the user stories that are pulled into a sprint. These stories have to be ready: clear, feasible, and testable. (You can learn more about decomposing epics into ready stories in my post "Epics and Ready Stories".)

Progressively refining your stories keeps your Product Canvas or backlog concise, it makes it easier to integrate new insights, and it reduces the effort required to stock your initial canvas or backlog. This is particularly valuable for new products and new features. Make sure you do not prescribe a solution in your stories. Rather focus on the "what", nor the "how". The latter is best captured in an architecture diagram.

Story Handoff

Some product owners diligently write user stories, and give them to the development team in the sprint planning meeting. This handoff is usually suboptimal, as it wastes the team's ideas and knowledge. Stories can hence be inappropriate, difficult to understand, unfeasible and not testable.

User stories, however, are not meant to be standalone documents. They should be complemented by a conversation between the product owner and the team, or even better: written collaboratively. A story wants to capture the essentials, and not specify every detail: The latter would be difficult for new features and too slow and too expensive in an agile / lean context.

User story writing should be a team effort, where product owner and development team create the stories together. Allocate time for a collaborative Product Canvas workshop or backlog grooming meeting, particularly when you develop a new product or new features. Trust me: Better stories and a better product will be your reward.

Criteria Crisis

Acceptance criteria are maybe the most misunderstood part of users stories. I have seen detailed stories with no acceptance criteria, criteria that restate the narrative, and criteria that hide new stories, or even contain entire workflows. The last two mistakes are exemplified by the following story (the narrative is on the card on the left, the "acceptance criteria" on the right):

The idea behind acceptance criteria is simple: They allow you to describe the conditions that have to be fulfilled so that the story is done, that it can be exposed to the users and the other stakeholders. This ensures that you gather feedback and/or release features, and it helps the team plan and track their work: The criteria enrich the story and make it more precise and testable. (The criteria all stories have to fulfil such as "online help is available" are not stated in the acceptance criteria but in the Definition of Done.)

As a rule of thumb, I like to work with three to five criteria per story, and I am not worried if my epics don't have acceptance criteria to start with. Ready stories, however, must provide meaningful criteria. You can find sample acceptance criteria in my posts "Epics and Ready Stories" and "Nonfunctional Requirements".

http://www.romanpichler.com/blog/user-stories/writing-good-user-stories/

10 Tips for Writing Good User Stories

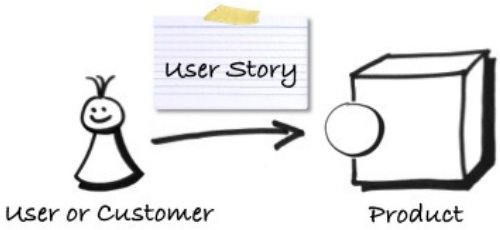
Posted on Thursday 7th October 2010

Summary

User stories are probably the most popular agile technique to capture product functionality: Working with user stories is easy. But writing good stories can be hard. The following ten tips help you create good stories.

1. Focus on the user

As its name suggests, a user story tells a story about a customer or user employing the product. Write therefore stories from the user's perspective and show how a user or customer uses some product functionality.

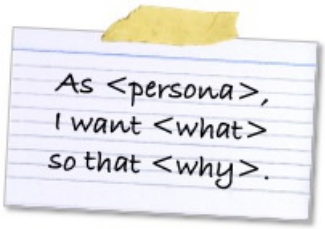


2. Write stories collaboratively

A user story is not a specification, but an communication and collaboration tool. Stories should not be handed off to the development team but be complemented by a conversation: Product owner and team should talk about stories, or even better, write them together: Invite the team to create and detail the stories together. This leverages the creativity and the knowledge of the team and usually results in better user stories.

3. Keep your stories simple and concise

Write your stories so that they are easy to understand. Keep them simple and concise. Avoid confusing and ambiguous terms, and use active voice. Focus on what's important, and leave out non-essential information. The following template puts the user or customer into the story and makes its benefit explicit. But use the template only if it is helpful. Try out different ways to write your stories to understand what works best for you.

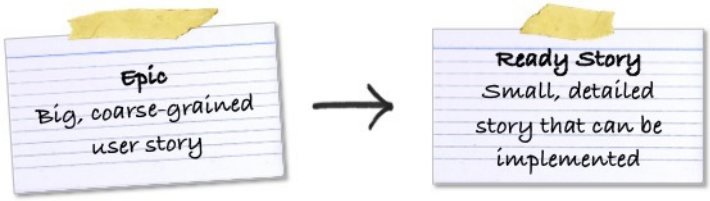


4. Start with epics

Epics are big, coarse-grained user stories. Starting with epics allows you to sketch the product functionality without committing to the details. This is particularly helpful for new products and new features, as it buys you time to learn more about the users and how to best meet their needs. It also reduces the time and effort required to integrate new insights: If you have lots of detailed stories, then it's often tricky to relate the feedback you receive to the right stories. I like to use the personas with their goals to discover the right epics.

5. Decompose your stories until they are ready

Break your epics into smaller, detailed stories until they are ready: clear, feasible, and testable. Everyone should have a shared understanding of the story's meaning; the story should not too big, and there has to be an effective way to determine if the story is done.



6. Add acceptance criteria

As you decompose epics into smaller stories, remember to add acceptance criteria. Acceptance criteria complement the story's narrative: They allow you to describe the conditions that have to be fulfilled so that the story is done. The criteria enrich the story and make it more precise and testable, and they ensures that the story can be demoed or released to the users and the other stakeholders.

7. Group user stories into themes

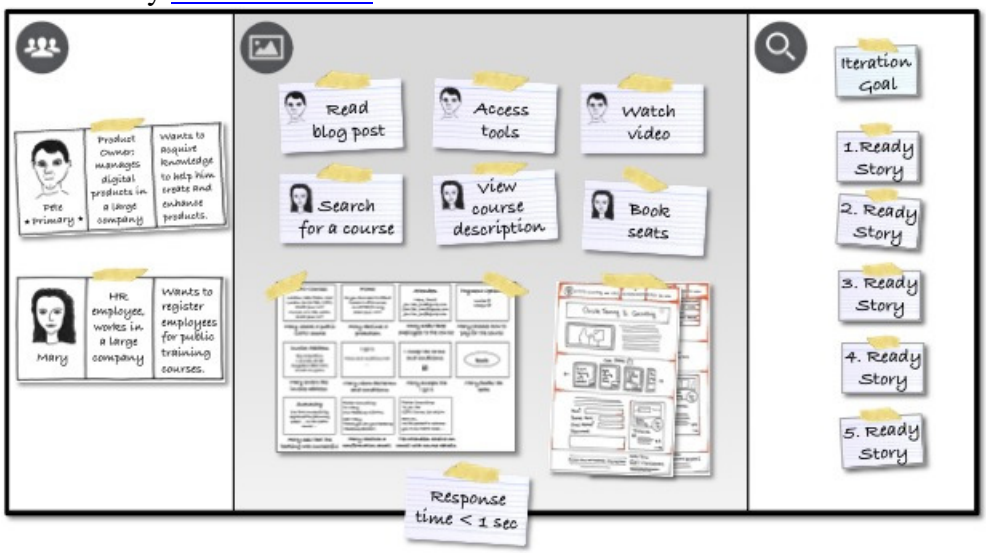
Using themes helps you organise your stories. Each theme is a group of related stories. Sample themes for a mobile phone are email, calendar, voice communication, and organizer, for instance. Themes make it easier to check for completeness and consistency, they structure your product canvas or product backlog, and they facilitate prioritisation.

8. Use paper cards

Paper cards are not only cheap and easy to use. They facilitate collaboration, as everyone can take a card and jot down an idea. Cards can also be easily grouped on the table or wall to check for consistency and completeness. Even if your stories are stored electronically, it is worthwhile to use paper cards when you write new stories.

9. Keep your stories visible and accessible

Stories want to communication information. Don't hide them on a network drive, the corporate intranet jungle, or a licensed tool. Make them visible instead, for instance, by putting them up on the wall. A great tool to discover, visualise, and manage your stories is my Product Canvas.



10. Don't solely rely on user stories

Creating a great user experience (UX) requires more than writing user stories. Also consider the user journeys and interactions, the visual design, and the nonfunctional properties of your product. This results in a holistic description that makes it easier to identify risks and assumptions, and it increases the chances of creating a product with the right user experience.

http://www.romanpichler.com/blog/user-stories/5-common-user-story-mistakes/

5 Common User Story Mistakes

Posted on Wednesday 12th June 2013

Summary

This post discusses five common user story mistakes and helps you overcome them.

User stories are a simple, yet effective way to communicate how a user or customer employs a product. But writing user stories that help a team build great software can be challenging. The following list summarises five common user story mistakes to help you improve your stories.

Story Mania

Some product owners and teams are so fond of user stories that everything is expressed as a story. This either results in some rather odd stories – stories that capture the user interface design, complex user interactions, and technical requirements; or these aspects are simply overlooked.

Like any technique, user story writing has its strengths and limitations. I find stories particularly well suited to capture product functionality, and when applied properly, nonfunctional requirements. But user interface design and complex user interactions are better described by other techniques including design sketches, mock-ups, scenarios, and storyboards. Complement your user stories therefore with other techniques, and don't feel obliged to only use stories.

User Incognito

A user story tells a story about a user interacting with the product. Some stories, however, omit the beneficiary altogether, or they talk about a mysterious user as in "As the user, I want to …" But if it's not clear who the user is, why should the story be developed? How can we be confident that the story will benefit someone?

Make sure that all your stories have a clear user or customer. As I like to work with personas, I use personas in my stories (instead of user roles). This connects each story to the right persona, and it allows me to understand if and to which extend the story addresses the persona's need or goal. To achieve this, I use the following template: As <persona>, I want <something> so that <benefit>.

Disastrous Details

The devil is in the details: Some stories are too big and vague for the team to understand and implement. Others contain too much detail, or even prescribe a solution. Getting the details right seems to be a battle the product owner can only loose.

My recommendation is: Start with big, coarse-grained stories called epics particularly for all new features. Epics allow you to capture an idea without committing to the details. Then break an epic into more detailed user stories. The new user stories replace the epic, and they provide more information about the product's functionality. Pay particular attention to the user stories that are pulled into a sprint. These stories have to be ready: clear, feasible, and testable. (You can learn more about decomposing epics into ready stories in my post "Epics and Ready Stories".)

Progressively refining your stories keeps your Product Canvas or backlog concise, it makes it easier to integrate new insights, and it reduces the effort required to stock your initial canvas or backlog. This is particularly valuable for new products and new features. Make sure you do not prescribe a solution in your stories. Rather focus on the "what", nor the "how". The latter is best captured in an architecture diagram.

Story Handoff

Some product owners diligently write user stories, and give them to the development team in the sprint planning meeting. This handoff is usually suboptimal, as it wastes the team's ideas and knowledge. Stories can hence be inappropriate, difficult to understand, unfeasible and not testable.

User stories, however, are not meant to be standalone documents. They should be complemented by a conversation between the product owner and the team, or even better: written collaboratively. A story wants to capture the essentials, and not specify every detail: The latter would be difficult for new features and too slow and too expensive in an agile / lean context.

User story writing should be a team effort, where product owner and development team create the stories together. Allocate time for a collaborative Product Canvas workshop or backlog grooming meeting, particularly when you develop a new product or new features. Trust me: Better stories and a better product will be your reward.

Criteria Crisis

Acceptance criteria are maybe the most misunderstood part of users stories. I have seen detailed stories with no acceptance criteria, criteria that restate the narrative, and criteria that hide new stories, or even contain entire workflows. The last two mistakes are exemplified by the following story (the narrative is on the card on the left, the "acceptance criteria" on the right):

The idea behind acceptance criteria is simple: They allow you to describe the conditions that have to be fulfilled so that the story is done, that it can be exposed to the users and the other stakeholders. This ensures that you gather feedback and/or release features, and it helps the team plan and track their work: The criteria enrich the story and make it more precise and testable. (The criteria all stories have to fulfil such as "online help is available" are not stated in the acceptance criteria but in the Definition of Done.)

As a rule of thumb, I like to work with three to five criteria per story, and I am not worried if my epics don't have acceptance criteria to start with. Ready stories, however, must provide meaningful criteria. You can find sample acceptance criteria in my posts "Epics and Ready Stories" and "Nonfunctional Requirements".

http://www.romanpichler.com/blog/user-stories/writing-good-user-stories/

10 Tips for Writing Good User Stories

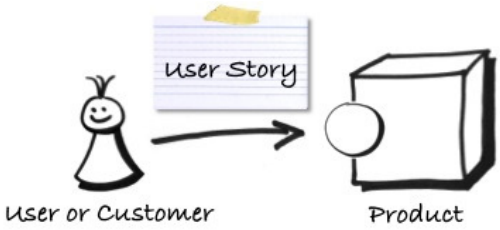
Posted on Thursday 7th October 2010

Summary

User stories are probably the most popular agile technique to capture product functionality: Working with user stories is easy. But writing good stories can be hard. The following ten tips help you create good stories.

1. Focus on the user

As its name suggests, a user story tells a story about a customer or user employing the product. Write therefore stories from the user's perspective and show how a user or customer uses some product functionality.

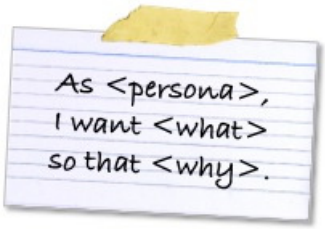


2. Write stories collaboratively

A user story is not a specification, but an communication and collaboration tool. Stories should not be handed off to the development team but be complemented by a conversation: Product owner and team should talk about stories, or even better, write them together: Invite the team to create and detail the stories together. This leverages the creativity and the knowledge of the team and usually results in better user stories.

3. Keep your stories simple and concise

Write your stories so that they are easy to understand. Keep them simple and concise. Avoid confusing and ambiguous terms, and use active voice. Focus on what's important, and leave out non-essential information. The following template puts the user or customer into the story and makes its benefit explicit. But use the template only if it is helpful. Try out different ways to write your stories to understand what works best for you.

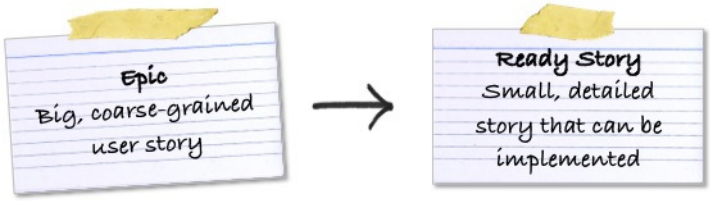


4. Start with epics

Epics are big, coarse-grained user stories. Starting with epics allows you to sketch the product functionality without committing to the details. This is particularly helpful for new products and new features, as it buys you time to learn more about the users and how to best meet their needs. It also reduces the time and effort required to integrate new insights: If you have lots of detailed stories, then it's often tricky to relate the feedback you receive to the right stories. I like to use the personas with their goals to discover the right epics.

5. Decompose your stories until they are ready

Break your epics into smaller, detailed stories until they are ready: clear, feasible, and testable. Everyone should have a shared understanding of the story's meaning; the story should not too big, and there has to be an effective way to determine if the story is done.



6. Add acceptance criteria

As you decompose epics into smaller stories, remember to add acceptance criteria. Acceptance criteria complement the story's narrative: They allow you to describe the conditions that have to be fulfilled so that the story is done. The criteria enrich the story and make it more precise and testable, and they ensures that the story can be demoed or released to the users and the other stakeholders.

7. Group user stories into themes

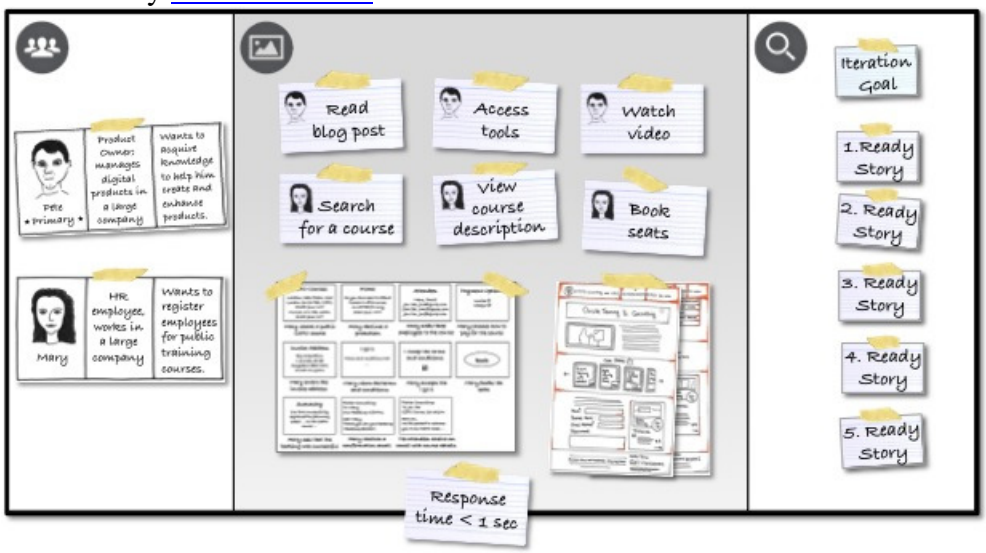
Using themes helps you organise your stories. Each theme is a group of related stories. Sample themes for a mobile phone are email, calendar, voice communication, and organizer, for instance. Themes make it easier to check for completeness and consistency, they structure your product canvas or product backlog, and they facilitate prioritisation.

8. Use paper cards

Paper cards are not only cheap and easy to use. They facilitate collaboration, as everyone can take a card and jot down an idea. Cards can also be easily grouped on the table or wall to check for consistency and completeness. Even if your stories are stored electronically, it is worthwhile to use paper cards when you write new stories.

9. Keep your stories visible and accessible

Stories want to communication information. Don't hide them on a network drive, the corporate intranet jungle, or a licensed tool. Make them visible instead, for instance, by putting them up on the wall. A great tool to discover, visualise, and manage your stories is my Product Canvas.



10. Don't solely rely on user stories

Creating a great user experience (UX) requires more than writing user stories. Also consider the user journeys and interactions, the visual design, and the nonfunctional properties of your product. This results in a holistic description that makes it easier to identify risks and assumptions, and it increases the chances of creating a product with the right user experience.

http://www.romanpichler.com/blog/user-stories/5-common-user-story-mistakes/

5 Common User Story Mistakes

Posted on Wednesday 12th June 2013

Summary

This post discusses five common user story mistakes and helps you overcome them.

User stories are a simple, yet effective way to communicate how a user or customer employs a product. But writing user stories that help a team build great software can be challenging. The following list summarises five common user story mistakes to help you improve your stories.

Story Mania

Some product owners and teams are so fond of user stories that everything is expressed as a story. This either results in some rather odd stories – stories that capture the user interface design, complex user interactions, and technical requirements; or these aspects are simply overlooked.

Like any technique, user story writing has its strengths and limitations. I find stories particularly well suited to capture product functionality, and when applied properly, nonfunctional requirements. But user interface design and complex user interactions are better described by other techniques including design sketches, mock-ups, scenarios, and storyboards. Complement your user stories therefore with other techniques, and don't feel obliged to only use stories.

User Incognito

A user story tells a story about a user interacting with the product. Some stories, however, omit the beneficiary altogether, or they talk about a mysterious user as in "As the user, I want to …" But if it's not clear who the user is, why should the story be developed? How can we be confident that the story will benefit someone?

Make sure that all your stories have a clear user or customer. As I like to work with personas, I use personas in my stories (instead of user roles). This connects each story to the right persona, and it allows me to understand if and to which extend the story addresses the persona's need or goal. To achieve this, I use the following template: As <persona>, I want <something> so that <benefit>.

Disastrous Details

The devil is in the details: Some stories are too big and vague for the team to understand and implement. Others contain too much detail, or even prescribe a solution. Getting the details right seems to be a battle the product owner can only loose.

My recommendation is: Start with big, coarse-grained stories called epics particularly for all new features. Epics allow you to capture an idea without committing to the details. Then break an epic into more detailed user stories. The new user stories replace the epic, and they provide more information about the product's functionality. Pay particular attention to the user stories that are pulled into a sprint. These stories have to be ready: clear, feasible, and testable. (You can learn more about decomposing epics into ready stories in my post "Epics and Ready Stories".)

Progressively refining your stories keeps your Product Canvas or backlog concise, it makes it easier to integrate new insights, and it reduces the effort required to stock your initial canvas or backlog. This is particularly valuable for new products and new features. Make sure you do not prescribe a solution in your stories. Rather focus on the "what", nor the "how". The latter is best captured in an architecture diagram.

Story Handoff

Some product owners diligently write user stories, and give them to the development team in the sprint planning meeting. This handoff is usually suboptimal, as it wastes the team's ideas and knowledge. Stories can hence be inappropriate, difficult to understand, unfeasible and not testable.

User stories, however, are not meant to be standalone documents. They should be complemented by a conversation between the product owner and the team, or even better: written collaboratively. A story wants to capture the essentials, and not specify every detail: The latter would be difficult for new features and too slow and too expensive in an agile / lean context.

User story writing should be a team effort, where product owner and development team create the stories together. Allocate time for a collaborative Product Canvas workshop or backlog grooming meeting, particularly when you develop a new product or new features. Trust me: Better stories and a better product will be your reward.

Criteria Crisis

Acceptance criteria are maybe the most misunderstood part of users stories. I have seen detailed stories with no acceptance criteria, criteria that restate the narrative, and criteria that hide new stories, or even contain entire workflows. The last two mistakes are exemplified by the following story (the narrative is on the card on the left, the "acceptance criteria" on the right):

The idea behind acceptance criteria is simple: They allow you to describe the conditions that have to be fulfilled so that the story is done, that it can be exposed to the users and the other stakeholders. This ensures that you gather feedback and/or release features, and it helps the team plan and track their work: The criteria enrich the story and make it more precise and testable. (The criteria all stories have to fulfil such as "online help is available" are not stated in the acceptance criteria but in the Definition of Done.)

As a rule of thumb, I like to work with three to five criteria per story, and I am not worried if my epics don't have acceptance criteria to start with. Ready stories, however, must provide meaningful criteria. You can find sample acceptance criteria in my posts "Epics and Ready Stories" and "Nonfunctional Requirements".

http://www.romanpichler.com/blog/user-stories/writing-good-user-stories/

10 Tips for Writing Good User Stories

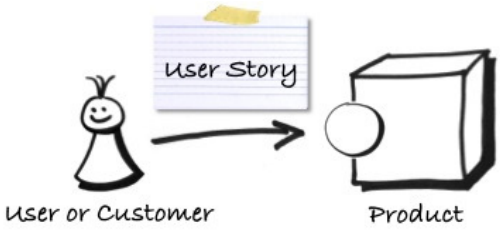
Posted on Thursday 7th October 2010

Summary

User stories are probably the most popular agile technique to capture product functionality: Working with user stories is easy. But writing good stories can be hard. The following ten tips help you create good stories.

1. Focus on the user

As its name suggests, a user story tells a story about a customer or user employing the product. Write therefore stories from the user's perspective and show how a user or customer uses some product functionality.

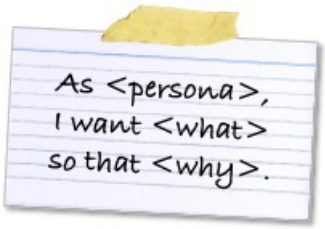


2. Write stories collaboratively

A user story is not a specification, but an communication and collaboration tool. Stories should not be handed off to the development team but be complemented by a conversation: Product owner and team should talk about stories, or even better, write them together: Invite the team to create and detail the stories together. This leverages the creativity and the knowledge of the team and usually results in better user stories.

3. Keep your stories simple and concise

Write your stories so that they are easy to understand. Keep them simple and concise. Avoid confusing and ambiguous terms, and use active voice. Focus on what's important, and leave out non-essential information. The following template puts the user or customer into the story and makes its benefit explicit. But use the template only if it is helpful. Try out different ways to write your stories to understand what works best for you.

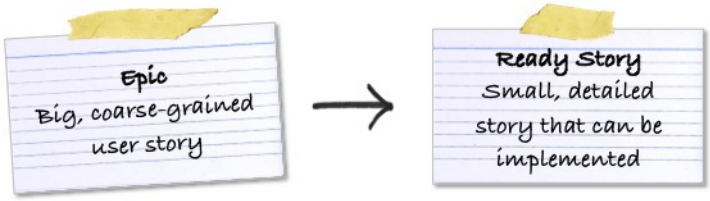


4. Start with epics

Epics are big, coarse-grained user stories. Starting with epics allows you to sketch the product functionality without committing to the details. This is particularly helpful for new products and new features, as it buys you time to learn more about the users and how to best meet their needs. It also reduces the time and effort required to integrate new insights: If you have lots of detailed stories, then it's often tricky to relate the feedback you receive to the right stories. I like to use the personas with their goals to discover the right epics.

5. Decompose your stories until they are ready

Break your epics into smaller, detailed stories until they are ready: clear, feasible, and testable. Everyone should have a shared understanding of the story's meaning; the story should not too big, and there has to be an effective way to determine if the story is done.



6. Add acceptance criteria

As you decompose epics into smaller stories, remember to add acceptance criteria. Acceptance criteria complement the story's narrative: They allow you to describe the conditions that have to be fulfilled so that the story is done. The criteria enrich the story and make it more precise and testable, and they ensures that the story can be demoed or released to the users and the other stakeholders.

7. Group user stories into themes

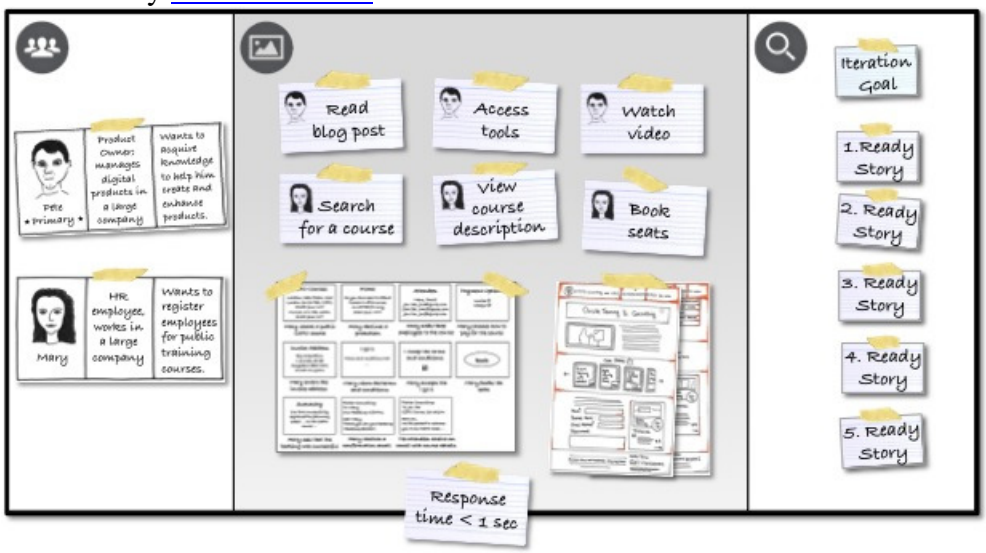
Using themes helps you organise your stories. Each theme is a group of related stories. Sample themes for a mobile phone are email, calendar, voice communication, and organizer, for instance. Themes make it easier to check for completeness and consistency, they structure your product canvas or product backlog, and they facilitate prioritisation.

8. Use paper cards

Paper cards are not only cheap and easy to use. They facilitate collaboration, as everyone can take a card and jot down an idea. Cards can also be easily grouped on the table or wall to check for consistency and completeness. Even if your stories are stored electronically, it is worthwhile to use paper cards when you write new stories.

9. Keep your stories visible and accessible

Stories want to communication information. Don't hide them on a network drive, the corporate intranet jungle, or a licensed tool. Make them visible instead, for instance, by putting them up on the wall. A great tool to discover, visualise, and manage your stories is my Product Canvas.



10. Don't solely rely on user stories

Creating a great user experience (UX) requires more than writing user stories. Also consider the user journeys and interactions, the visual design, and the nonfunctional properties of your product. This results in a holistic description that makes it easier to identify risks and assumptions, and it increases the chances of creating a product with the right user experience.

http://www.romanpichler.com/blog/user-stories/5-common-user-story-mistakes/

5 Common User Story Mistakes

Posted on Wednesday 12th June 2013

Summary

This post discusses five common user story mistakes and helps you overcome them.

User stories are a simple, yet effective way to communicate how a user or customer employs a product. But writing user stories that help a team build great software can be challenging. The following list summarises five common user story mistakes to help you improve your stories.

Story Mania

Some product owners and teams are so fond of user stories that everything is expressed as a story. This either results in some rather odd stories – stories that capture the user interface design, complex user interactions, and technical requirements; or these aspects are simply overlooked.

Like any technique, user story writing has its strengths and limitations. I find stories particularly well suited to capture product functionality, and when applied properly, nonfunctional requirements. But user interface design and complex user interactions are better described by other techniques including design sketches, mock-ups, scenarios, and storyboards. Complement your user stories therefore with other techniques, and don't feel obliged to only use stories.

User Incognito

A user story tells a story about a user interacting with the product. Some stories, however, omit the beneficiary altogether, or they talk about a mysterious user as in "As the user, I want to …" But if it's not clear who the user is, why should the story be developed? How can we be confident that the story will benefit someone?

Make sure that all your stories have a clear user or customer. As I like to work with personas, I use personas in my stories (instead of user roles). This connects each story to the right persona, and it allows me to understand if and to which extend the story addresses the persona's need or goal. To achieve this, I use the following template: As <persona>, I want <something> so that <benefit>.

Disastrous Details

The devil is in the details: Some stories are too big and vague for the team to understand and implement. Others contain too much detail, or even prescribe a solution. Getting the details right seems to be a battle the product owner can only loose.

My recommendation is: Start with big, coarse-grained stories called epics particularly for all new features. Epics allow you to capture an idea without committing to the details. Then break an epic into more detailed user stories. The new user stories replace the epic, and they provide more information about the product's functionality. Pay particular attention to the user stories that are pulled into a sprint. These stories have to be ready: clear, feasible, and testable. (You can learn more about decomposing epics into ready stories in my post "Epics and Ready Stories".)

Progressively refining your stories keeps your Product Canvas or backlog concise, it makes it easier to integrate new insights, and it reduces the effort required to stock your initial canvas or backlog. This is particularly valuable for new products and new features. Make sure you do not prescribe a solution in your stories. Rather focus on the "what", nor the "how". The latter is best captured in an architecture diagram.

Story Handoff

Some product owners diligently write user stories, and give them to the development team in the sprint planning meeting. This handoff is usually suboptimal, as it wastes the team's ideas and knowledge. Stories can hence be inappropriate, difficult to understand, unfeasible and not testable.

User stories, however, are not meant to be standalone documents. They should be complemented by a conversation between the product owner and the team, or even better: written collaboratively. A story wants to capture the essentials, and not specify every detail: The latter would be difficult for new features and too slow and too expensive in an agile / lean context.

User story writing should be a team effort, where product owner and development team create the stories together. Allocate time for a collaborative Product Canvas workshop or backlog grooming meeting, particularly when you develop a new product or new features. Trust me: Better stories and a better product will be your reward.

Criteria Crisis

Acceptance criteria are maybe the most misunderstood part of users stories. I have seen detailed stories with no acceptance criteria, criteria that restate the narrative, and criteria that hide new stories, or even contain entire workflows. The last two mistakes are exemplified by the following story (the narrative is on the card on the left, the "acceptance criteria" on the right):

The idea behind acceptance criteria is simple: They allow you to describe the conditions that have to be fulfilled so that the story is done, that it can be exposed to the users and the other stakeholders. This ensures that you gather feedback and/or release features, and it helps the team plan and track their work: The criteria enrich the story and make it more precise and testable. (The criteria all stories have to fulfil such as "online help is available" are not stated in the acceptance criteria but in the Definition of Done.)

As a rule of thumb, I like to work with three to five criteria per story, and I am not worried if my epics don't have acceptance criteria to start with. Ready stories, however, must provide meaningful criteria. You can find sample acceptance criteria in my posts "Epics and Ready Stories" and "Nonfunctional Requirements".

http://www.romanpichler.com/blog/user-stories/writing-good-user-stories/

10 Tips for Writing Good User Stories

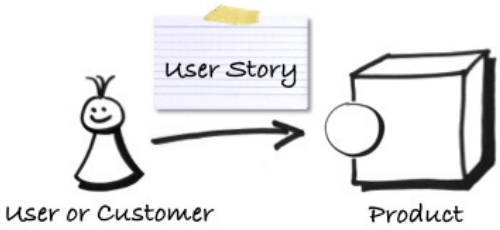
Posted on Thursday 7th October 2010

Summary

User stories are probably the most popular agile technique to capture product functionality: Working with user stories is easy. But writing good stories can be hard. The following ten tips help you create good stories.

1. Focus on the user

As its name suggests, a user story tells a story about a customer or user employing the product. Write therefore stories from the user's perspective and show how a user or customer uses some product functionality.

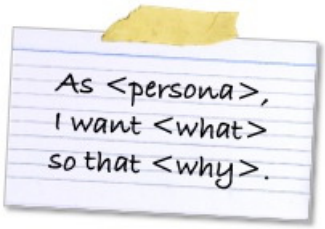


2. Write stories collaboratively

A user story is not a specification, but an communication and collaboration tool. Stories should not be handed off to the development team but be complemented by a conversation: Product owner and team should talk about stories, or even better, write them together: Invite the team to create and detail the stories together. This leverages the creativity and the knowledge of the team and usually results in better user stories.

3. Keep your stories simple and concise

Write your stories so that they are easy to understand. Keep them simple and concise. Avoid confusing and ambiguous terms, and use active voice. Focus on what's important, and leave out non-essential information. The following template puts the user or customer into the story and makes its benefit explicit. But use the template only if it is helpful. Try out different ways to write your stories to understand what works best for you.

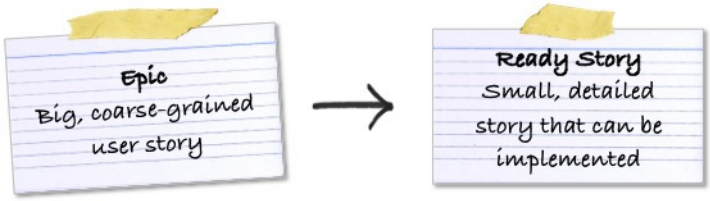


4. Start with epics

Epics are big, coarse-grained user stories. Starting with epics allows you to sketch the product functionality without committing to the details. This is particularly helpful for new products and new features, as it buys you time to learn more about the users and how to best meet their needs. It also reduces the time and effort required to integrate new insights: If you have lots of detailed stories, then it's often tricky to relate the feedback you receive to the right stories. I like to use the personas with their goals to discover the right epics.

5. Decompose your stories until they are ready

Break your epics into smaller, detailed stories until they are ready: clear, feasible, and testable. Everyone should have a shared understanding of the story's meaning; the story should not too big, and there has to be an effective way to determine if the story is done.



6. Add acceptance criteria

As you decompose epics into smaller stories, remember to add acceptance criteria. Acceptance criteria complement the story's narrative: They allow you to describe the conditions that have to be fulfilled so that the story is done. The criteria enrich the story and make it more precise and testable, and they ensures that the story can be demoed or released to the users and the other stakeholders.

7. Group user stories into themes

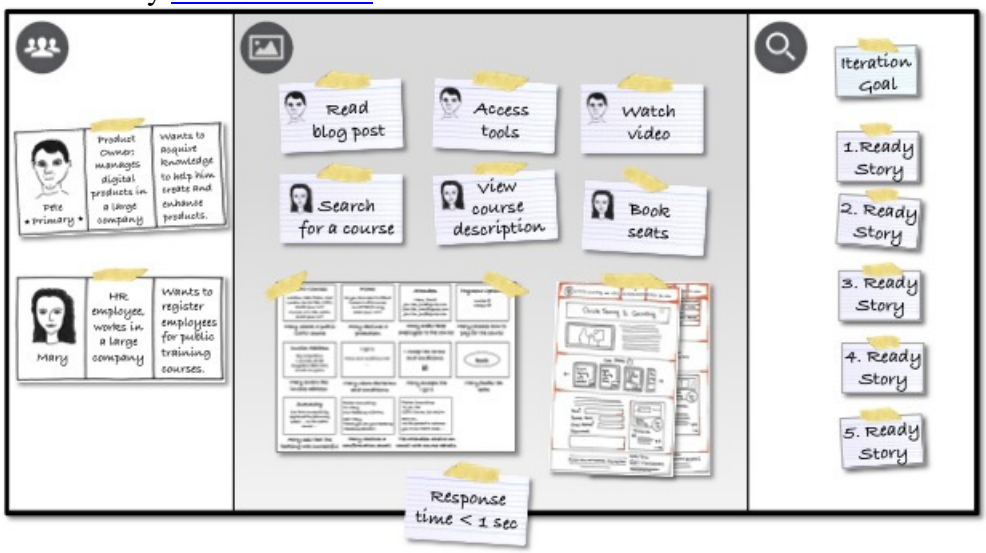
Using themes helps you organise your stories. Each theme is a group of related stories. Sample themes for a mobile phone are email, calendar, voice communication, and organizer, for instance. Themes make it easier to check for completeness and consistency, they structure your product canvas or product backlog, and they facilitate prioritisation.

8. Use paper cards

Paper cards are not only cheap and easy to use. They facilitate collaboration, as everyone can take a card and jot down an idea. Cards can also be easily grouped on the table or wall to check for consistency and completeness. Even if your stories are stored electronically, it is worthwhile to use paper cards when you write new stories.

9. Keep your stories visible and accessible

Stories want to communication information. Don't hide them on a network drive, the corporate intranet jungle, or a licensed tool. Make them visible instead, for instance, by putting them up on the wall. A great tool to discover, visualise, and manage your stories is my Product Canvas.



10. Don't solely rely on user stories

Creating a great user experience (UX) requires more than writing user stories. Also consider the user journeys and interactions, the visual design, and the nonfunctional properties of your product. This results in a holistic description that makes it easier to identify risks and assumptions, and it increases the chances of creating a product with the right user experience.

http://www.romanpichler.com/blog/user-stories/5-common-user-story-mistakes/

5 Common User Story Mistakes

Posted on Wednesday 12th June 2013

Summary

This post discusses five common user story mistakes and helps you overcome them.

User stories are a simple, yet effective way to communicate how a user or customer employs a product. But writing user stories that help a team build great software can be challenging. The following list summarises five common user story mistakes to help you improve your stories.

Story Mania

Some product owners and teams are so fond of user stories that everything is expressed as a story. This either results in some rather odd stories – stories that capture the user interface design, complex user interactions, and technical requirements; or these aspects are simply overlooked.

Like any technique, user story writing has its strengths and limitations. I find stories particularly well suited to capture product functionality, and when applied properly, nonfunctional requirements. But user interface design and complex user interactions are better described by other techniques including design sketches, mock-ups, scenarios, and storyboards. Complement your user stories therefore with other techniques, and don't feel obliged to only use stories.

User Incognito

A user story tells a story about a user interacting with the product. Some stories, however, omit the beneficiary altogether, or they talk about a mysterious user as in "As the user, I want to …" But if it's not clear who the user is, why should the story be developed? How can we be confident that the story will benefit someone?

Make sure that all your stories have a clear user or customer. As I like to work with personas, I use personas in my stories (instead of user roles). This connects each story to the right persona, and it allows me to understand if and to which extend the story addresses the persona's need or goal. To achieve this, I use the following template: As <persona>, I want <something> so that <benefit>.

Disastrous Details

The devil is in the details: Some stories are too big and vague for the team to understand and implement. Others contain too much detail, or even prescribe a solution. Getting the details right seems to be a battle the product owner can only loose.

My recommendation is: Start with big, coarse-grained stories called epics particularly for all new features. Epics allow you to capture an idea without committing to the details. Then break an epic into more detailed user stories. The new user stories replace the epic, and they provide more information about the product's functionality. Pay particular attention to the user stories that are pulled into a sprint. These stories have to be ready: clear, feasible, and testable. (You can learn more about decomposing epics into ready stories in my post "Epics and Ready Stories".)

Progressively refining your stories keeps your Product Canvas or backlog concise, it makes it easier to integrate new insights, and it reduces the effort required to stock your initial canvas or backlog. This is particularly valuable for new products and new features. Make sure you do not prescribe a solution in your stories. Rather focus on the "what", nor the "how". The latter is best captured in an architecture diagram.

Story Handoff

Some product owners diligently write user stories, and give them to the development team in the sprint planning meeting. This handoff is usually suboptimal, as it wastes the team's ideas and knowledge. Stories can hence be inappropriate, difficult to understand, unfeasible and not testable.

User stories, however, are not meant to be standalone documents. They should be complemented by a conversation between the product owner and the team, or even better: written collaboratively. A story wants to capture the essentials, and not specify every detail: The latter would be difficult for new features and too slow and too expensive in an agile / lean context.

User story writing should be a team effort, where product owner and development team create the stories together. Allocate time for a collaborative Product Canvas workshop or backlog grooming meeting, particularly when you develop a new product or new features. Trust me: Better stories and a better product will be your reward.

Criteria Crisis

Acceptance criteria are maybe the most misunderstood part of users stories. I have seen detailed stories with no acceptance criteria, criteria that restate the narrative, and criteria that hide new stories, or even contain entire workflows. The last two mistakes are exemplified by the following story (the narrative is on the card on the left, the "acceptance criteria" on the right):

The idea behind acceptance criteria is simple: They allow you to describe the conditions that have to be fulfilled so that the story is done, that it can be exposed to the users and the other stakeholders. This ensures that you gather feedback and/or release features, and it helps the team plan and track their work: The criteria enrich the story and make it more precise and testable. (The criteria all stories have to fulfil such as "online help is available" are not stated in the acceptance criteria but in the Definition of Done.)

As a rule of thumb, I like to work with three to five criteria per story, and I am not worried if my epics don't have acceptance criteria to start with. Ready stories, however, must provide meaningful criteria. You can find sample acceptance criteria in my posts "Epics and Ready Stories" and "Nonfunctional Requirements".

http://www.romanpichler.com/blog/user-stories/writing-good-user-stories/

10 Tips for Writing Good User Stories

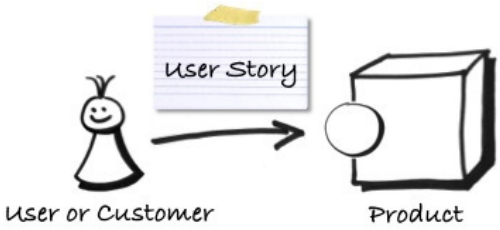
Posted on Thursday 7th October 2010

Summary

User stories are probably the most popular agile technique to capture product functionality: Working with user stories is easy. But writing good stories can be hard. The following ten tips help you create good stories.

1. Focus on the user

As its name suggests, a user story tells a story about a customer or user employing the product. Write therefore stories from the user's perspective and show how a user or customer uses some product functionality.

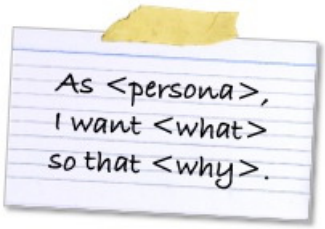


2. Write stories collaboratively

A user story is not a specification, but an communication and collaboration tool. Stories should not be handed off to the development team but be complemented by a conversation: Product owner and team should talk about stories, or even better, write them together: Invite the team to create and detail the stories together. This leverages the creativity and the knowledge of the team and usually results in better user stories.

3. Keep your stories simple and concise

Write your stories so that they are easy to understand. Keep them simple and concise. Avoid confusing and ambiguous terms, and use active voice. Focus on what's important, and leave out non-essential information. The following template puts the user or customer into the story and makes its benefit explicit. But use the template only if it is helpful. Try out different ways to write your stories to understand what works best for you.

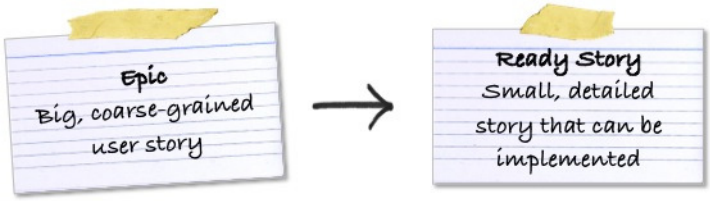


4. Start with epics

Epics are big, coarse-grained user stories. Starting with epics allows you to sketch the product functionality without committing to the details. This is particularly helpful for new products and new features, as it buys you time to learn more about the users and how to best meet their needs. It also reduces the time and effort required to integrate new insights: If you have lots of detailed stories, then it's often tricky to relate the feedback you receive to the right stories. I like to use the personas with their goals to discover the right epics.

5. Decompose your stories until they are ready

Break your epics into smaller, detailed stories until they are ready: clear, feasible, and testable. Everyone should have a shared understanding of the story's meaning; the story should not too big, and there has to be an effective way to determine if the story is done.



6. Add acceptance criteria

As you decompose epics into smaller stories, remember to add acceptance criteria. Acceptance criteria complement the story's narrative: They allow you to describe the conditions that have to be fulfilled so that the story is done. The criteria enrich the story and make it more precise and testable, and they ensures that the story can be demoed or released to the users and the other stakeholders.

7. Group user stories into themes

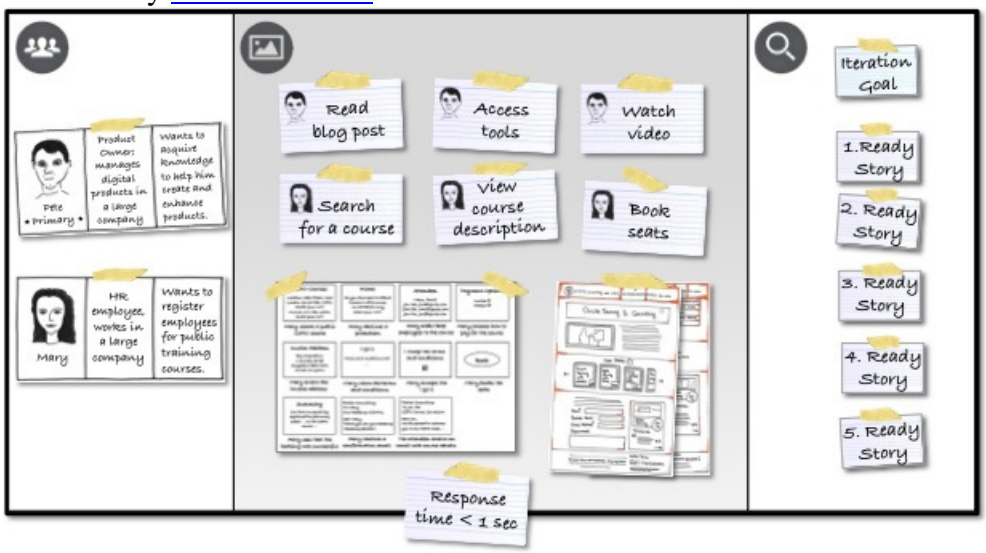
Using themes helps you organise your stories. Each theme is a group of related stories. Sample themes for a mobile phone are email, calendar, voice communication, and organizer, for instance. Themes make it easier to check for completeness and consistency, they structure your product canvas or product backlog, and they facilitate prioritisation.

8. Use paper cards

Paper cards are not only cheap and easy to use. They facilitate collaboration, as everyone can take a card and jot down an idea. Cards can also be easily grouped on the table or wall to check for consistency and completeness. Even if your stories are stored electronically, it is worthwhile to use paper cards when you write new stories.

9. Keep your stories visible and accessible

Stories want to communication information. Don't hide them on a network drive, the corporate intranet jungle, or a licensed tool. Make them visible instead, for instance, by putting them up on the wall. A great tool to discover, visualise, and manage your stories is my Product Canvas.



10. Don't solely rely on user stories

Creating a great user experience (UX) requires more than writing user stories. Also consider the user journeys and interactions, the visual design, and the nonfunctional properties of your product. This results in a holistic description that makes it easier to identify risks and assumptions, and it increases the chances of creating a product with the right user experience.

http://www.romanpichler.com/blog/user-stories/5-common-user-story-mistakes/

5 Common User Story Mistakes

Posted on Wednesday 12th June 2013

Summary

This post discusses five common user story mistakes and helps you overcome them.

User stories are a simple, yet effective way to communicate how a user or customer employs a product. But writing user stories that help a team build great software can be challenging. The following list summarises five common user story mistakes to help you improve your stories.

Story Mania

Some product owners and teams are so fond of user stories that everything is expressed as a story. This either results in some rather odd stories – stories that capture the user interface design, complex user interactions, and technical requirements; or these aspects are simply overlooked.

Like any technique, user story writing has its strengths and limitations. I find stories particularly well suited to capture product functionality, and when applied properly, nonfunctional requirements. But user interface design and complex user interactions are better described by other techniques including design sketches, mock-ups, scenarios, and storyboards. Complement your user stories therefore with other techniques, and don't feel obliged to only use stories.

User Incognito

A user story tells a story about a user interacting with the product. Some stories, however, omit the beneficiary altogether, or they talk about a mysterious user as in "As the user, I want to …" But if it's not clear who the user is, why should the story be developed? How can we be confident that the story will benefit someone?

Make sure that all your stories have a clear user or customer. As I like to work with personas, I use personas in my stories (instead of user roles). This connects each story to the right persona, and it allows me to understand if and to which extend the story addresses the persona's need or goal. To achieve this, I use the following template: As <persona>, I want <something> so that <benefit>.

Disastrous Details

The devil is in the details: Some stories are too big and vague for the team to understand and implement. Others contain too much detail, or even prescribe a solution. Getting the details right seems to be a battle the product owner can only loose.

My recommendation is: Start with big, coarse-grained stories called epics particularly for all new features. Epics allow you to capture an idea without committing to the details. Then break an epic into more detailed user stories. The new user stories replace the epic, and they provide more information about the product's functionality. Pay particular attention to the user stories that are pulled into a sprint. These stories have to be ready: clear, feasible, and testable. (You can learn more about decomposing epics into ready stories in my post "Epics and Ready Stories".)

Progressively refining your stories keeps your Product Canvas or backlog concise, it makes it easier to integrate new insights, and it reduces the effort required to stock your initial canvas or backlog. This is particularly valuable for new products and new features. Make sure you do not prescribe a solution in your stories. Rather focus on the "what", nor the "how". The latter is best captured in an architecture diagram.

Story Handoff

Some product owners diligently write user stories, and give them to the development team in the sprint planning meeting. This handoff is usually suboptimal, as it wastes the team's ideas and knowledge. Stories can hence be inappropriate, difficult to understand, unfeasible and not testable.

User stories, however, are not meant to be standalone documents. They should be complemented by a conversation between the product owner and the team, or even better: written collaboratively. A story wants to capture the essentials, and not specify every detail: The latter would be difficult for new features and too slow and too expensive in an agile / lean context.

User story writing should be a team effort, where product owner and development team create the stories together. Allocate time for a collaborative Product Canvas workshop or backlog grooming meeting, particularly when you develop a new product or new features. Trust me: Better stories and a better product will be your reward.

Criteria Crisis

Acceptance criteria are maybe the most misunderstood part of users stories. I have seen detailed stories with no acceptance criteria, criteria that restate the narrative, and criteria that hide new stories, or even contain entire workflows. The last two mistakes are exemplified by the following story (the narrative is on the card on the left, the "acceptance criteria" on the right):

The idea behind acceptance criteria is simple: They allow you to describe the conditions that have to be fulfilled so that the story is done, that it can be exposed to the users and the other stakeholders. This ensures that you gather feedback and/or release features, and it helps the team plan and track their work: The criteria enrich the story and make it more precise and testable. (The criteria all stories have to fulfil such as "online help is available" are not stated in the acceptance criteria but in the Definition of Done.)

As a rule of thumb, I like to work with three to five criteria per story, and I am not worried if my epics don't have acceptance criteria to start with. Ready stories, however, must provide meaningful criteria. You can find sample acceptance criteria in my posts "Epics and Ready Stories" and "Nonfunctional Requirements".

http://www.romanpichler.com/blog/user-stories/writing-good-user-stories/

10 Tips for Writing Good User Stories

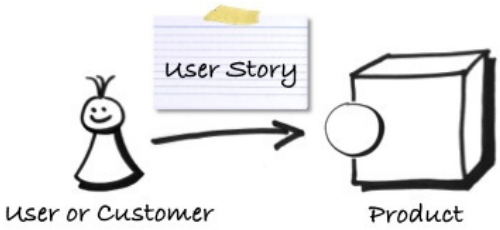
Posted on Thursday 7th October 2010

Summary

User stories are probably the most popular agile technique to capture product functionality: Working with user stories is easy. But writing good stories can be hard. The following ten tips help you create good stories.

1. Focus on the user

As its name suggests, a user story tells a story about a customer or user employing the product. Write therefore stories from the user's perspective and show how a user or customer uses some product functionality.

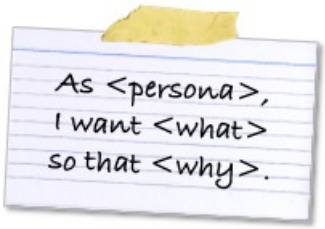


2. Write stories collaboratively

A user story is not a specification, but an communication and collaboration tool. Stories should not be handed off to the development team but be complemented by a conversation: Product owner and team should talk about stories, or even better, write them together: Invite the team to create and detail the stories together. This leverages the creativity and the knowledge of the team and usually results in better user stories.

3. Keep your stories simple and concise

Write your stories so that they are easy to understand. Keep them simple and concise. Avoid confusing and ambiguous terms, and use active voice. Focus on what's important, and leave out non-essential information. The following template puts the user or customer into the story and makes its benefit explicit. But use the template only if it is helpful. Try out different ways to write your stories to understand what works best for you.

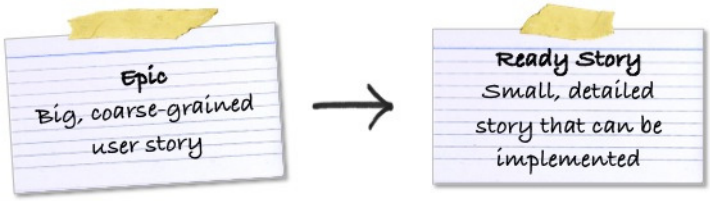


4. Start with epics

Epics are big, coarse-grained user stories. Starting with epics allows you to sketch the product functionality without committing to the details. This is particularly helpful for new products and new features, as it buys you time to learn more about the users and how to best meet their needs. It also reduces the time and effort required to integrate new insights: If you have lots of detailed stories, then it's often tricky to relate the feedback you receive to the right stories. I like to use the personas with their goals to discover the right epics.

5. Decompose your stories until they are ready

Break your epics into smaller, detailed stories until they are ready: clear, feasible, and testable. Everyone should have a shared understanding of the story's meaning; the story should not too big, and there has to be an effective way to determine if the story is done.



6. Add acceptance criteria

As you decompose epics into smaller stories, remember to add acceptance criteria. Acceptance criteria complement the story's narrative: They allow you to describe the conditions that have to be fulfilled so that the story is done. The criteria enrich the story and make it more precise and testable, and they ensures that the story can be demoed or released to the users and the other stakeholders.

7. Group user stories into themes

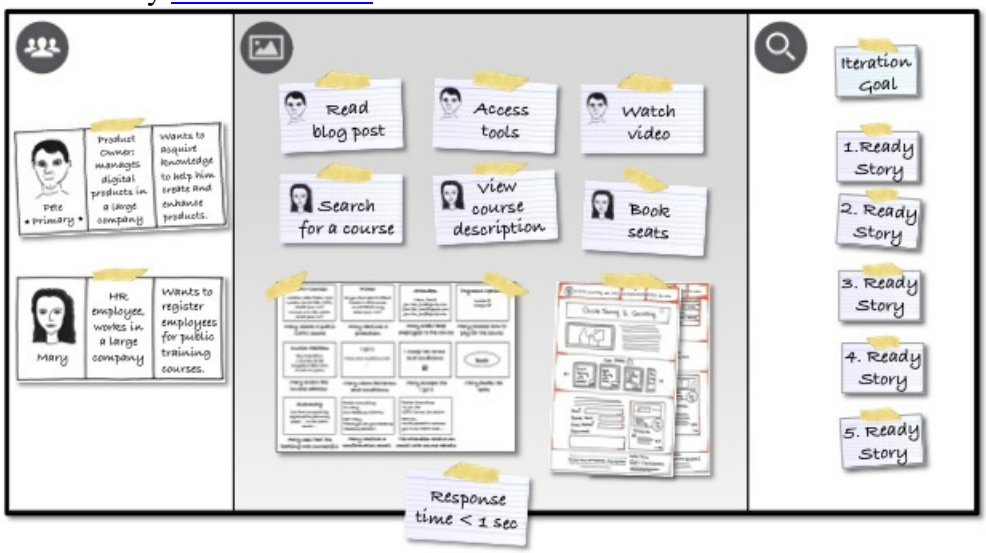
Using themes helps you organise your stories. Each theme is a group of related stories. Sample themes for a mobile phone are email, calendar, voice communication, and organizer, for instance. Themes make it easier to check for completeness and consistency, they structure your product canvas or product backlog, and they facilitate prioritisation.

8. Use paper cards

Paper cards are not only cheap and easy to use. They facilitate collaboration, as everyone can take a card and jot down an idea. Cards can also be easily grouped on the table or wall to check for consistency and completeness. Even if your stories are stored electronically, it is worthwhile to use paper cards when you write new stories.

9. Keep your stories visible and accessible

Stories want to communication information. Don't hide them on a network drive, the corporate intranet jungle, or a licensed tool. Make them visible instead, for instance, by putting them up on the wall. A great tool to discover, visualise, and manage your stories is my Product Canvas.



10. Don't solely rely on user stories

Creating a great user experience (UX) requires more than writing user stories. Also consider the user journeys and interactions, the visual design, and the nonfunctional properties of your product. This results in a holistic description that makes it easier to identify risks and assumptions, and it increases the chances of creating a product with the right user experience.

http://www.romanpichler.com/blog/user-stories/5-common-user-story-mistakes/

5 Common User Story Mistakes

Posted on Wednesday 12th June 2013

Summary

This post discusses five common user story mistakes and helps you overcome them.

User stories are a simple, yet effective way to communicate how a user or customer employs a product. But writing user stories that help a team build great software can be challenging. The following list summarises five common user story mistakes to help you improve your stories.

Story Mania

Some product owners and teams are so fond of user stories that everything is expressed as a story. This either results in some rather odd stories – stories that capture the user interface design, complex user interactions, and technical requirements; or these aspects are simply overlooked.

Like any technique, user story writing has its strengths and limitations. I find stories particularly well suited to capture product functionality, and when applied properly, nonfunctional requirements. But user interface design and complex user interactions are better described by other techniques including design sketches, mock-ups, scenarios, and storyboards. Complement your user stories therefore with other techniques, and don't feel obliged to only use stories.

User Incognito

A user story tells a story about a user interacting with the product. Some stories, however, omit the beneficiary altogether, or they talk about a mysterious user as in "As the user, I want to …" But if it's not clear who the user is, why should the story be developed? How can we be confident that the story will benefit someone?

Make sure that all your stories have a clear user or customer. As I like to work with personas, I use personas in my stories (instead of user roles). This connects each story to the right persona, and it allows me to understand if and to which extend the story addresses the persona's need or goal. To achieve this, I use the following template: As <persona>, I want <something> so that <benefit>.

Disastrous Details

The devil is in the details: Some stories are too big and vague for the team to understand and implement. Others contain too much detail, or even prescribe a solution. Getting the details right seems to be a battle the product owner can only loose.

My recommendation is: Start with big, coarse-grained stories called epics particularly for all new features. Epics allow you to capture an idea without committing to the details. Then break an epic into more detailed user stories. The new user stories replace the epic, and they provide more information about the product's functionality. Pay particular attention to the user stories that are pulled into a sprint. These stories have to be ready: clear, feasible, and testable. (You can learn more about decomposing epics into ready stories in my post "Epics and Ready Stories".)

Progressively refining your stories keeps your Product Canvas or backlog concise, it makes it easier to integrate new insights, and it reduces the effort required to stock your initial canvas or backlog. This is particularly valuable for new products and new features. Make sure you do not prescribe a solution in your stories. Rather focus on the "what", nor the "how". The latter is best captured in an architecture diagram.

Story Handoff

Some product owners diligently write user stories, and give them to the development team in the sprint planning meeting. This handoff is usually suboptimal, as it wastes the team's ideas and knowledge. Stories can hence be inappropriate, difficult to understand, unfeasible and not testable.

User stories, however, are not meant to be standalone documents. They should be complemented by a conversation between the product owner and the team, or even better: written collaboratively. A story wants to capture the essentials, and not specify every detail: The latter would be difficult for new features and too slow and too expensive in an agile / lean context.

User story writing should be a team effort, where product owner and development team create the stories together. Allocate time for a collaborative Product Canvas workshop or backlog grooming meeting, particularly when you develop a new product or new features. Trust me: Better stories and a better product will be your reward.

Criteria Crisis

Acceptance criteria are maybe the most misunderstood part of users stories. I have seen detailed stories with no acceptance criteria, criteria that restate the narrative, and criteria that hide new stories, or even contain entire workflows. The last two mistakes are exemplified by the following story (the narrative is on the card on the left, the "acceptance criteria" on the right):

The idea behind acceptance criteria is simple: They allow you to describe the conditions that have to be fulfilled so that the story is done, that it can be exposed to the users and the other stakeholders. This ensures that you gather feedback and/or release features, and it helps the team plan and track their work: The criteria enrich the story and make it more precise and testable. (The criteria all stories have to fulfil such as "online help is available" are not stated in the acceptance criteria but in the Definition of Done.)

As a rule of thumb, I like to work with three to five criteria per story, and I am not worried if my epics don't have acceptance criteria to start with. Ready stories, however, must provide meaningful criteria. You can find sample acceptance criteria in my posts "Epics and Ready Stories" and "Nonfunctional Requirements".

http://www.romanpichler.com/blog/user-stories/writing-good-user-stories/

10 Tips for Writing Good User Stories

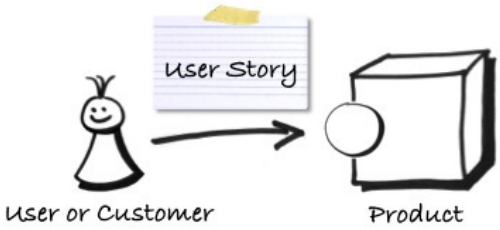
Posted on Thursday 7th October 2010

Summary

User stories are probably the most popular agile technique to capture product functionality: Working with user stories is easy. But writing good stories can be hard. The following ten tips help you create good stories.

1. Focus on the user

As its name suggests, a user story tells a story about a customer or user employing the product. Write therefore stories from the user's perspective and show how a user or customer uses some product functionality.

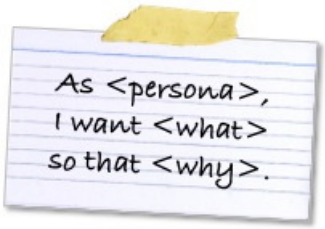


2. Write stories collaboratively

A user story is not a specification, but an communication and collaboration tool. Stories should not be handed off to the development team but be complemented by a conversation: Product owner and team should talk about stories, or even better, write them together: Invite the team to create and detail the stories together. This leverages the creativity and the knowledge of the team and usually results in better user stories.

3. Keep your stories simple and concise

Write your stories so that they are easy to understand. Keep them simple and concise. Avoid confusing and ambiguous terms, and use active voice. Focus on what's important, and leave out non-essential information. The following template puts the user or customer into the story and makes its benefit explicit. But use the template only if it is helpful. Try out different ways to write your stories to understand what works best for you.

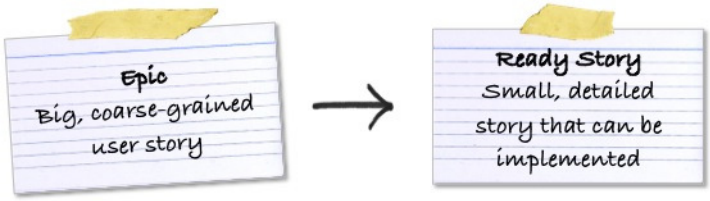


4. Start with epics

Epics are big, coarse-grained user stories. Starting with epics allows you to sketch the product functionality without committing to the details. This is particularly helpful for new products and new features, as it buys you time to learn more about the users and how to best meet their needs. It also reduces the time and effort required to integrate new insights: If you have lots of detailed stories, then it's often tricky to relate the feedback you receive to the right stories. I like to use the personas with their goals to discover the right epics.

5. Decompose your stories until they are ready

Break your epics into smaller, detailed stories until they are ready: clear, feasible, and testable. Everyone should have a shared understanding of the story's meaning; the story should not too big, and there has to be an effective way to determine if the story is done.



6. Add acceptance criteria

As you decompose epics into smaller stories, remember to add acceptance criteria. Acceptance criteria complement the story's narrative: They allow you to describe the conditions that have to be fulfilled so that the story is done. The criteria enrich the story and make it more precise and testable, and they ensures that the story can be demoed or released to the users and the other stakeholders.

7. Group user stories into themes

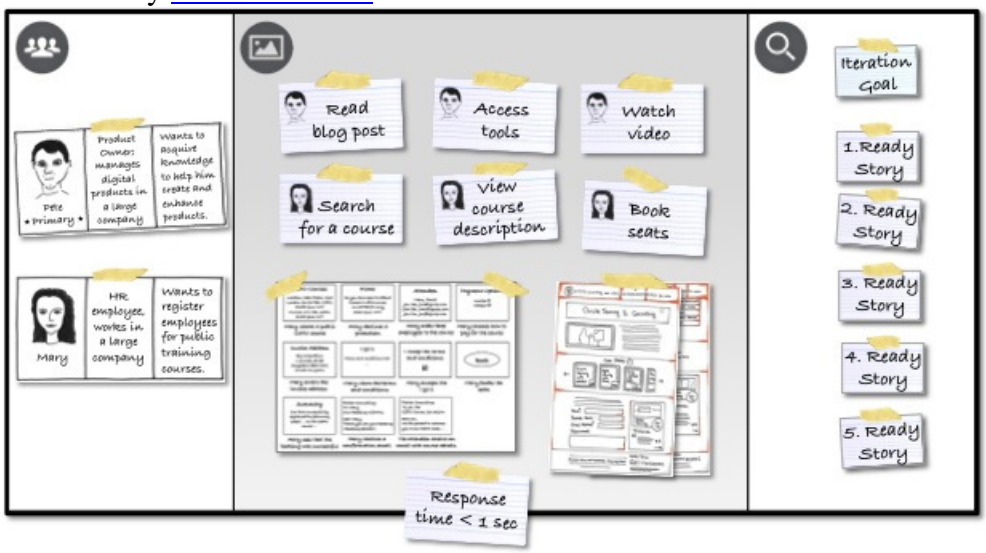
Using themes helps you organise your stories. Each theme is a group of related stories. Sample themes for a mobile phone are email, calendar, voice communication, and organizer, for instance. Themes make it easier to check for completeness and consistency, they structure your product canvas or product backlog, and they facilitate prioritisation.

8. Use paper cards

Paper cards are not only cheap and easy to use. They facilitate collaboration, as everyone can take a card and jot down an idea. Cards can also be easily grouped on the table or wall to check for consistency and completeness. Even if your stories are stored electronically, it is worthwhile to use paper cards when you write new stories.

9. Keep your stories visible and accessible

Stories want to communication information. Don't hide them on a network drive, the corporate intranet jungle, or a licensed tool. Make them visible instead, for instance, by putting them up on the wall. A great tool to discover, visualise, and manage your stories is my Product Canvas.



10. Don't solely rely on user stories

Creating a great user experience (UX) requires more than writing user stories. Also consider the user journeys and interactions, the visual design, and the nonfunctional properties of your product. This results in a holistic description that makes it easier to identify risks and assumptions, and it increases the chances of creating a product with the right user experience.

http://www.romanpichler.com/blog/user-stories/5-common-user-story-mistakes/

5 Common User Story Mistakes

Posted on Wednesday 12th June 2013

Summary

This post discusses five common user story mistakes and helps you overcome them.

User stories are a simple, yet effective way to communicate how a user or customer employs a product. But writing user stories that help a team build great software can be challenging. The following list summarises five common user story mistakes to help you improve your stories.

Story Mania

Some product owners and teams are so fond of user stories that everything is expressed as a story. This either results in some rather odd stories – stories that capture the user interface design, complex user interactions, and technical requirements; or these aspects are simply overlooked.

Like any technique, user story writing has its strengths and limitations. I find stories particularly well suited to capture product functionality, and when applied properly, nonfunctional requirements. But user interface design and complex user interactions are better described by other techniques including design sketches, mock-ups, scenarios, and storyboards. Complement your user stories therefore with other techniques, and don't feel obliged to only use stories.

User Incognito

A user story tells a story about a user interacting with the product. Some stories, however, omit the beneficiary altogether, or they talk about a mysterious user as in "As the user, I want to …" But if it's not clear who the user is, why should the story be developed? How can we be confident that the story will benefit someone?

Make sure that all your stories have a clear user or customer. As I like to work with personas, I use personas in my stories (instead of user roles). This connects each story to the right persona, and it allows me to understand if and to which extend the story addresses the persona's need or goal. To achieve this, I use the following template: As <persona>, I want <something> so that <benefit>.

Disastrous Details

The devil is in the details: Some stories are too big and vague for the team to understand and implement. Others contain too much detail, or even prescribe a solution. Getting the details right seems to be a battle the product owner can only loose.

My recommendation is: Start with big, coarse-grained stories called epics particularly for all new features. Epics allow you to capture an idea without committing to the details. Then break an epic into more detailed user stories. The new user stories replace the epic, and they provide more information about the product's functionality. Pay particular attention to the user stories that are pulled into a sprint. These stories have to be ready: clear, feasible, and testable. (You can learn more about decomposing epics into ready stories in my post "Epics and Ready Stories".)

Progressively refining your stories keeps your Product Canvas or backlog concise, it makes it easier to integrate new insights, and it reduces the effort required to stock your initial canvas or backlog. This is particularly valuable for new products and new features. Make sure you do not prescribe a solution in your stories. Rather focus on the "what", nor the "how". The latter is best captured in an architecture diagram.

Story Handoff

Some product owners diligently write user stories, and give them to the development team in the sprint planning meeting. This handoff is usually suboptimal, as it wastes the team's ideas and knowledge. Stories can hence be inappropriate, difficult to understand, unfeasible and not testable.

User stories, however, are not meant to be standalone documents. They should be complemented by a conversation between the product owner and the team, or even better: written collaboratively. A story wants to capture the essentials, and not specify every detail: The latter would be difficult for new features and too slow and too expensive in an agile / lean context.

User story writing should be a team effort, where product owner and development team create the stories together. Allocate time for a collaborative Product Canvas workshop or backlog grooming meeting, particularly when you develop a new product or new features. Trust me: Better stories and a better product will be your reward.

Criteria Crisis

Acceptance criteria are maybe the most misunderstood part of users stories. I have seen detailed stories with no acceptance criteria, criteria that restate the narrative, and criteria that hide new stories, or even contain entire workflows. The last two mistakes are exemplified by the following story (the narrative is on the card on the left, the "acceptance criteria" on the right):

The idea behind acceptance criteria is simple: They allow you to describe the conditions that have to be fulfilled so that the story is done, that it can be exposed to the users and the other stakeholders. This ensures that you gather feedback and/or release features, and it helps the team plan and track their work: The criteria enrich the story and make it more precise and testable. (The criteria all stories have to fulfil such as "online help is available" are not stated in the acceptance criteria but in the Definition of Done.)

As a rule of thumb, I like to work with three to five criteria per story, and I am not worried if my epics don't have acceptance criteria to start with. Ready stories, however, must provide meaningful criteria. You can find sample acceptance criteria in my posts "Epics and Ready Stories" and "Nonfunctional Requirements".

http://www.romanpichler.com/blog/user-stories/writing-good-user-stories/

10 Tips for Writing Good User Stories

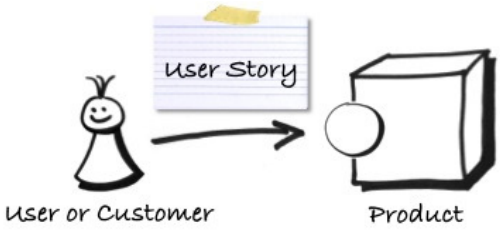
Posted on Thursday 7th October 2010

Summary

User stories are probably the most popular agile technique to capture product functionality: Working with user stories is easy. But writing good stories can be hard. The following ten tips help you create good stories.

1. Focus on the user

As its name suggests, a user story tells a story about a customer or user employing the product. Write therefore stories from the user's perspective and show how a user or customer uses some product functionality.

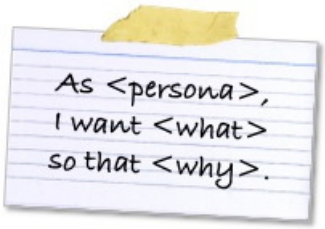


2. Write stories collaboratively

A user story is not a specification, but an communication and collaboration tool. Stories should not be handed off to the development team but be complemented by a conversation: Product owner and team should talk about stories, or even better, write them together: Invite the team to create and detail the stories together. This leverages the creativity and the knowledge of the team and usually results in better user stories.

3. Keep your stories simple and concise

Write your stories so that they are easy to understand. Keep them simple and concise. Avoid confusing and ambiguous terms, and use active voice. Focus on what's important, and leave out non-essential information. The following template puts the user or customer into the story and makes its benefit explicit. But use the template only if it is helpful. Try out different ways to write your stories to understand what works best for you.

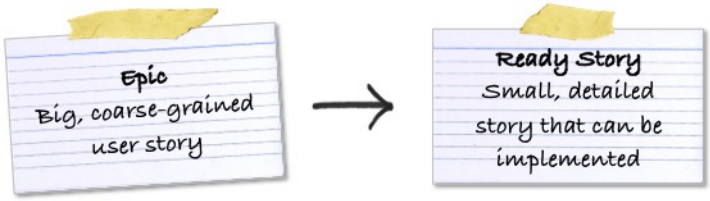


4. Start with epics

Epics are big, coarse-grained user stories. Starting with epics allows you to sketch the product functionality without committing to the details. This is particularly helpful for new products and new features, as it buys you time to learn more about the users and how to best meet their needs. It also reduces the time and effort required to integrate new insights: If you have lots of detailed stories, then it's often tricky to relate the feedback you receive to the right stories. I like to use the personas with their goals to discover the right epics.

5. Decompose your stories until they are ready

Break your epics into smaller, detailed stories until they are ready: clear, feasible, and testable. Everyone should have a shared understanding of the story's meaning; the story should not too big, and there has to be an effective way to determine if the story is done.



6. Add acceptance criteria

As you decompose epics into smaller stories, remember to add acceptance criteria. Acceptance criteria complement the story's narrative: They allow you to describe the conditions that have to be fulfilled so that the story is done. The criteria enrich the story and make it more precise and testable, and they ensures that the story can be demoed or released to the users and the other stakeholders.

7. Group user stories into themes

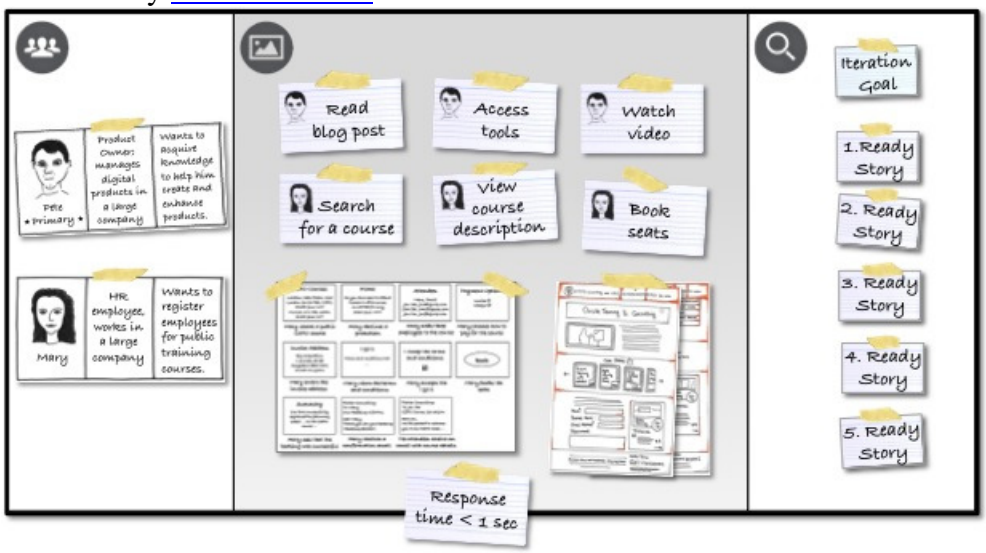
Using themes helps you organise your stories. Each theme is a group of related stories. Sample themes for a mobile phone are email, calendar, voice communication, and organizer, for instance. Themes make it easier to check for completeness and consistency, they structure your product canvas or product backlog, and they facilitate prioritisation.

8. Use paper cards

Paper cards are not only cheap and easy to use. They facilitate collaboration, as everyone can take a card and jot down an idea. Cards can also be easily grouped on the table or wall to check for consistency and completeness. Even if your stories are stored electronically, it is worthwhile to use paper cards when you write new stories.

9. Keep your stories visible and accessible

Stories want to communication information. Don't hide them on a network drive, the corporate intranet jungle, or a licensed tool. Make them visible instead, for instance, by putting them up on the wall. A great tool to discover, visualise, and manage your stories is my Product Canvas.



10. Don't solely rely on user stories

Creating a great user experience (UX) requires more than writing user stories. Also consider the user journeys and interactions, the visual design, and the nonfunctional properties of your product. This results in a holistic description that makes it easier to identify risks and assumptions, and it increases the chances of creating a product with the right user experience.

http://www.romanpichler.com/blog/user-stories/5-common-user-story-mistakes/

5 Common User Story Mistakes

Posted on Wednesday 12th June 2013

Summary

This post discusses five common user story mistakes and helps you overcome them.

User stories are a simple, yet effective way to communicate how a user or customer employs a product. But writing user stories that help a team build great software can be challenging. The following list summarises five common user story mistakes to help you improve your stories.

Story Mania

Some product owners and teams are so fond of user stories that everything is expressed as a story. This either results in some rather odd stories – stories that capture the user interface design, complex user interactions, and technical requirements; or these aspects are simply overlooked.

Like any technique, user story writing has its strengths and limitations. I find stories particularly well suited to capture product functionality, and when applied properly, nonfunctional requirements. But user interface design and complex user interactions are better described by other techniques including design sketches, mock-ups, scenarios, and storyboards. Complement your user stories therefore with other techniques, and don't feel obliged to only use stories.

User Incognito

A user story tells a story about a user interacting with the product. Some stories, however, omit the beneficiary altogether, or they talk about a mysterious user as in "As the user, I want to …" But if it's not clear who the user is, why should the story be developed? How can we be confident that the story will benefit someone?

Make sure that all your stories have a clear user or customer. As I like to work with personas, I use personas in my stories (instead of user roles). This connects each story to the right persona, and it allows me to understand if and to which extend the story addresses the persona's need or goal. To achieve this, I use the following template: As <persona>, I want <something> so that <benefit>.

Disastrous Details

The devil is in the details: Some stories are too big and vague for the team to understand and implement. Others contain too much detail, or even prescribe a solution. Getting the details right seems to be a battle the product owner can only loose.

My recommendation is: Start with big, coarse-grained stories called epics particularly for all new features. Epics allow you to capture an idea without committing to the details. Then break an epic into more detailed user stories. The new user stories replace the epic, and they provide more information about the product's functionality. Pay particular attention to the user stories that are pulled into a sprint. These stories have to be ready: clear, feasible, and testable. (You can learn more about decomposing epics into ready stories in my post "Epics and Ready Stories".)

Progressively refining your stories keeps your Product Canvas or backlog concise, it makes it easier to integrate new insights, and it reduces the effort required to stock your initial canvas or backlog. This is particularly valuable for new products and new features. Make sure you do not prescribe a solution in your stories. Rather focus on the "what", nor the "how". The latter is best captured in an architecture diagram.

Story Handoff

Some product owners diligently write user stories, and give them to the development team in the sprint planning meeting. This handoff is usually suboptimal, as it wastes the team's ideas and knowledge. Stories can hence be inappropriate, difficult to understand, unfeasible and not testable.

User stories, however, are not meant to be standalone documents. They should be complemented by a conversation between the product owner and the team, or even better: written collaboratively. A story wants to capture the essentials, and not specify every detail: The latter would be difficult for new features and too slow and too expensive in an agile / lean context.

User story writing should be a team effort, where product owner and development team create the stories together. Allocate time for a collaborative Product Canvas workshop or backlog grooming meeting, particularly when you develop a new product or new features. Trust me: Better stories and a better product will be your reward.

Criteria Crisis

Acceptance criteria are maybe the most misunderstood part of users stories. I have seen detailed stories with no acceptance criteria, criteria that restate the narrative, and criteria that hide new stories, or even contain entire workflows. The last two mistakes are exemplified by the following story (the narrative is on the card on the left, the "acceptance criteria" on the right):

The idea behind acceptance criteria is simple: They allow you to describe the conditions that have to be fulfilled so that the story is done, that it can be exposed to the users and the other stakeholders. This ensures that you gather feedback and/or release features, and it helps the team plan and track their work: The criteria enrich the story and make it more precise and testable. (The criteria all stories have to fulfil such as "online help is available" are not stated in the acceptance criteria but in the Definition of Done.)

As a rule of thumb, I like to work with three to five criteria per story, and I am not worried if my epics don't have acceptance criteria to start with. Ready stories, however, must provide meaningful criteria. You can find sample acceptance criteria in my posts "Epics and Ready Stories" and "Nonfunctional Requirements".

http://www.romanpichler.com/blog/user-stories/writing-good-user-stories/

10 Tips for Writing Good User Stories

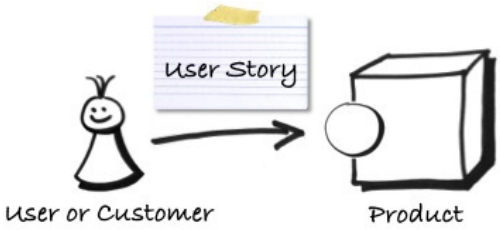
Posted on Thursday 7th October 2010

Summary

User stories are probably the most popular agile technique to capture product functionality: Working with user stories is easy. But writing good stories can be hard. The following ten tips help you create good stories.

1. Focus on the user

As its name suggests, a user story tells a story about a customer or user employing the product. Write therefore stories from the user's perspective and show how a user or customer uses some product functionality.

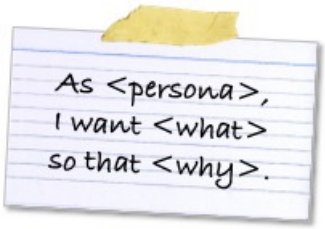


2. Write stories collaboratively

A user story is not a specification, but an communication and collaboration tool. Stories should not be handed off to the development team but be complemented by a conversation: Product owner and team should talk about stories, or even better, write them together: Invite the team to create and detail the stories together. This leverages the creativity and the knowledge of the team and usually results in better user stories.

3. Keep your stories simple and concise

Write your stories so that they are easy to understand. Keep them simple and concise. Avoid confusing and ambiguous terms, and use active voice. Focus on what's important, and leave out non-essential information. The following template puts the user or customer into the story and makes its benefit explicit. But use the template only if it is helpful. Try out different ways to write your stories to understand what works best for you.

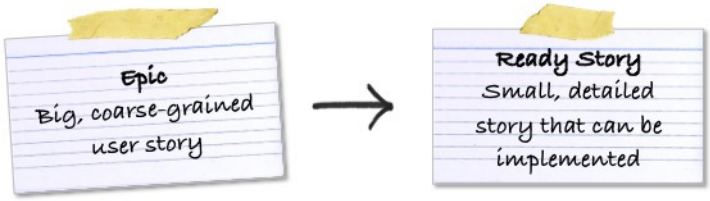


4. Start with epics

Epics are big, coarse-grained user stories. Starting with epics allows you to sketch the product functionality without committing to the details. This is particularly helpful for new products and new features, as it buys you time to learn more about the users and how to best meet their needs. It also reduces the time and effort required to integrate new insights: If you have lots of detailed stories, then it's often tricky to relate the feedback you receive to the right stories. I like to use the personas with their goals to discover the right epics.

5. Decompose your stories until they are ready

Break your epics into smaller, detailed stories until they are ready: clear, feasible, and testable. Everyone should have a shared understanding of the story's meaning; the story should not too big, and there has to be an effective way to determine if the story is done.



6. Add acceptance criteria

As you decompose epics into smaller stories, remember to add acceptance criteria. Acceptance criteria complement the story's narrative: They allow you to describe the conditions that have to be fulfilled so that the story is done. The criteria enrich the story and make it more precise and testable, and they ensures that the story can be demoed or released to the users and the other stakeholders.

7. Group user stories into themes

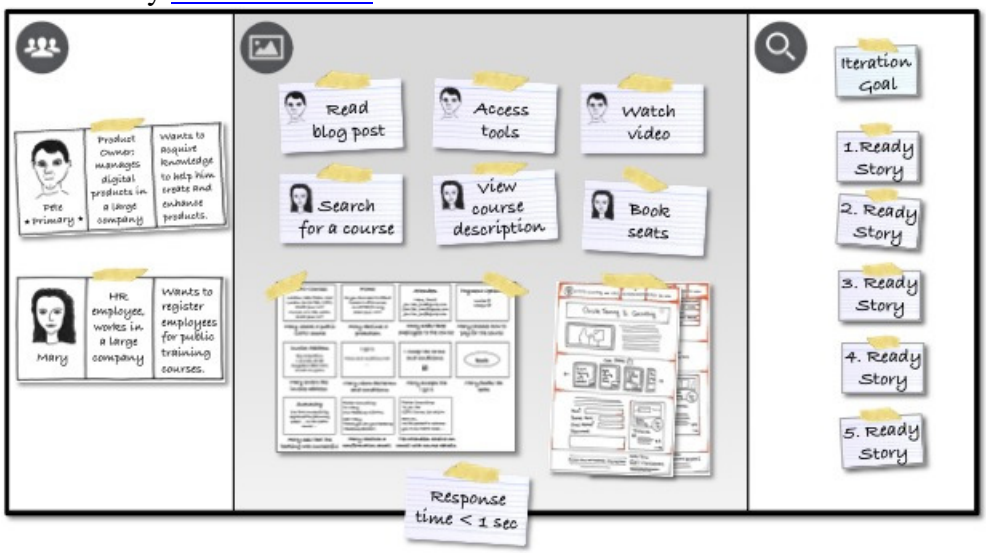
Using themes helps you organise your stories. Each theme is a group of related stories. Sample themes for a mobile phone are email, calendar, voice communication, and organizer, for instance. Themes make it easier to check for completeness and consistency, they structure your product canvas or product backlog, and they facilitate prioritisation.

8. Use paper cards

Paper cards are not only cheap and easy to use. They facilitate collaboration, as everyone can take a card and jot down an idea. Cards can also be easily grouped on the table or wall to check for consistency and completeness. Even if your stories are stored electronically, it is worthwhile to use paper cards when you write new stories.

9. Keep your stories visible and accessible

Stories want to communication information. Don't hide them on a network drive, the corporate intranet jungle, or a licensed tool. Make them visible instead, for instance, by putting them up on the wall. A great tool to discover, visualise, and manage your stories is my Product Canvas.



10. Don't solely rely on user stories

Creating a great user experience (UX) requires more than writing user stories. Also consider the user journeys and interactions, the visual design, and the nonfunctional properties of your product. This results in a holistic description that makes it easier to identify risks and assumptions, and it increases the chances of creating a product with the right user experience.

http://www.romanpichler.com/blog/user-stories/5-common-user-story-mistakes/

5 Common User Story Mistakes

Posted on Wednesday 12th June 2013

Summary

This post discusses five common user story mistakes and helps you overcome them.

User stories are a simple, yet effective way to communicate how a user or customer employs a product. But writing user stories that help a team build great software can be challenging. The following list summarises five common user story mistakes to help you improve your stories.

Story Mania

Some product owners and teams are so fond of user stories that everything is expressed as a story. This either results in some rather odd stories – stories that capture the user interface design, complex user interactions, and technical requirements; or these aspects are simply overlooked.

Like any technique, user story writing has its strengths and limitations. I find stories particularly well suited to capture product functionality, and when applied properly, nonfunctional requirements. But user interface design and complex user interactions are better described by other techniques including design sketches, mock-ups, scenarios, and storyboards. Complement your user stories therefore with other techniques, and don't feel obliged to only use stories.

User Incognito

A user story tells a story about a user interacting with the product. Some stories, however, omit the beneficiary altogether, or they talk about a mysterious user as in "As the user, I want to …" But if it's not clear who the user is, why should the story be developed? How can we be confident that the story will benefit someone?

Make sure that all your stories have a clear user or customer. As I like to work with personas, I use personas in my stories (instead of user roles). This connects each story to the right persona, and it allows me to understand if and to which extend the story addresses the persona's need or goal. To achieve this, I use the following template: As <persona>, I want <something> so that <benefit>.

Disastrous Details

The devil is in the details: Some stories are too big and vague for the team to understand and implement. Others contain too much detail, or even prescribe a solution. Getting the details right seems to be a battle the product owner can only loose.

My recommendation is: Start with big, coarse-grained stories called epics particularly for all new features. Epics allow you to capture an idea without committing to the details. Then break an epic into more detailed user stories. The new user stories replace the epic, and they provide more information about the product's functionality. Pay particular attention to the user stories that are pulled into a sprint. These stories have to be ready: clear, feasible, and testable. (You can learn more about decomposing epics into ready stories in my post "Epics and Ready Stories".)

Progressively refining your stories keeps your Product Canvas or backlog concise, it makes it easier to integrate new insights, and it reduces the effort required to stock your initial canvas or backlog. This is particularly valuable for new products and new features. Make sure you do not prescribe a solution in your stories. Rather focus on the "what", nor the "how". The latter is best captured in an architecture diagram.

Story Handoff

Some product owners diligently write user stories, and give them to the development team in the sprint planning meeting. This handoff is usually suboptimal, as it wastes the team's ideas and knowledge. Stories can hence be inappropriate, difficult to understand, unfeasible and not testable.

User stories, however, are not meant to be standalone documents. They should be complemented by a conversation between the product owner and the team, or even better: written collaboratively. A story wants to capture the essentials, and not specify every detail: The latter would be difficult for new features and too slow and too expensive in an agile / lean context.

User story writing should be a team effort, where product owner and development team create the stories together. Allocate time for a collaborative Product Canvas workshop or backlog grooming meeting, particularly when you develop a new product or new features. Trust me: Better stories and a better product will be your reward.

Criteria Crisis

Acceptance criteria are maybe the most misunderstood part of users stories. I have seen detailed stories with no acceptance criteria, criteria that restate the narrative, and criteria that hide new stories, or even contain entire workflows. The last two mistakes are exemplified by the following story (the narrative is on the card on the left, the "acceptance criteria" on the right):

The idea behind acceptance criteria is simple: They allow you to describe the conditions that have to be fulfilled so that the story is done, that it can be exposed to the users and the other stakeholders. This ensures that you gather feedback and/or release features, and it helps the team plan and track their work: The criteria enrich the story and make it more precise and testable. (The criteria all stories have to fulfil such as "online help is available" are not stated in the acceptance criteria but in the Definition of Done.)

As a rule of thumb, I like to work with three to five criteria per story, and I am not worried if my epics don't have acceptance criteria to start with. Ready stories, however, must provide meaningful criteria. You can find sample acceptance criteria in my posts "Epics and Ready Stories" and "Nonfunctional Requirements".

http://www.romanpichler.com/blog/user-stories/writing-good-user-stories/

10 Tips for Writing Good User Stories

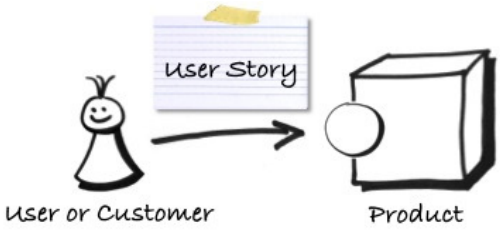
Posted on Thursday 7th October 2010

Summary

User stories are probably the most popular agile technique to capture product functionality: Working with user stories is easy. But writing good stories can be hard. The following ten tips help you create good stories.

1. Focus on the user

As its name suggests, a user story tells a story about a customer or user employing the product. Write therefore stories from the user's perspective and show how a user or customer uses some product functionality.

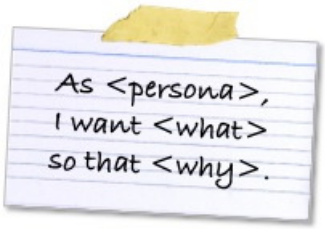


2. Write stories collaboratively

A user story is not a specification, but an communication and collaboration tool. Stories should not be handed off to the development team but be complemented by a conversation: Product owner and team should talk about stories, or even better, write them together: Invite the team to create and detail the stories together. This leverages the creativity and the knowledge of the team and usually results in better user stories.

3. Keep your stories simple and concise

Write your stories so that they are easy to understand. Keep them simple and concise. Avoid confusing and ambiguous terms, and use active voice. Focus on what's important, and leave out non-essential information. The following template puts the user or customer into the story and makes its benefit explicit. But use the template only if it is helpful. Try out different ways to write your stories to understand what works best for you.

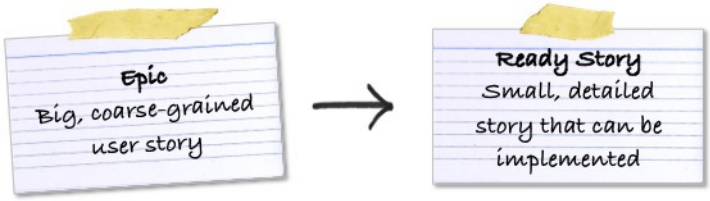


4. Start with epics

Epics are big, coarse-grained user stories. Starting with epics allows you to sketch the product functionality without committing to the details. This is particularly helpful for new products and new features, as it buys you time to learn more about the users and how to best meet their needs. It also reduces the time and effort required to integrate new insights: If you have lots of detailed stories, then it's often tricky to relate the feedback you receive to the right stories. I like to use the personas with their goals to discover the right epics.

5. Decompose your stories until they are ready

Break your epics into smaller, detailed stories until they are ready: clear, feasible, and testable. Everyone should have a shared understanding of the story's meaning; the story should not too big, and there has to be an effective way to determine if the story is done.



6. Add acceptance criteria

As you decompose epics into smaller stories, remember to add acceptance criteria. Acceptance criteria complement the story's narrative: They allow you to describe the conditions that have to be fulfilled so that the story is done. The criteria enrich the story and make it more precise and testable, and they ensures that the story can be demoed or released to the users and the other stakeholders.

7. Group user stories into themes

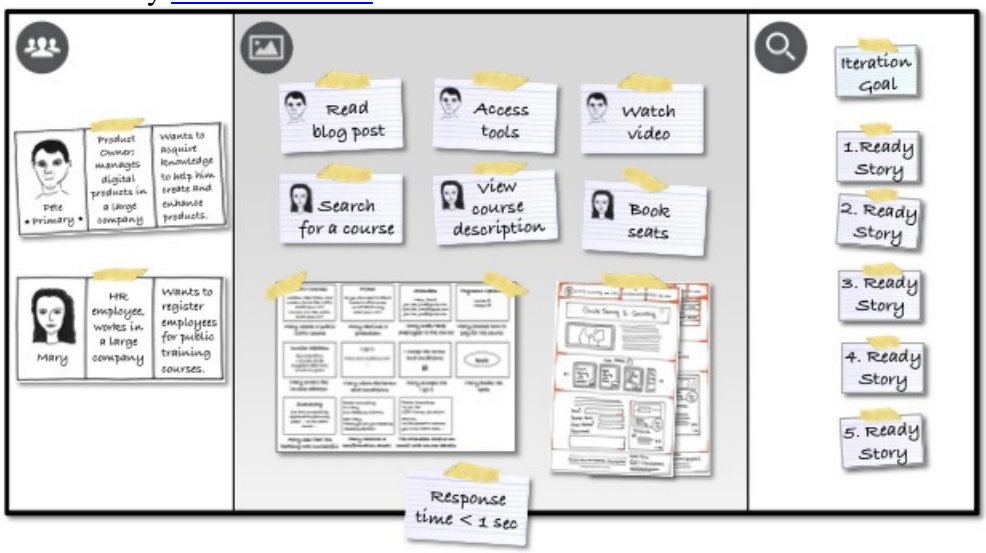
Using themes helps you organise your stories. Each theme is a group of related stories. Sample themes for a mobile phone are email, calendar, voice communication, and organizer, for instance. Themes make it easier to check for completeness and consistency, they structure your product canvas or product backlog, and they facilitate prioritisation.

8. Use paper cards

Paper cards are not only cheap and easy to use. They facilitate collaboration, as everyone can take a card and jot down an idea. Cards can also be easily grouped on the table or wall to check for consistency and completeness. Even if your stories are stored electronically, it is worthwhile to use paper cards when you write new stories.

9. Keep your stories visible and accessible

Stories want to communication information. Don't hide them on a network drive, the corporate intranet jungle, or a licensed tool. Make them visible instead, for instance, by putting them up on the wall. A great tool to discover, visualise, and manage your stories is my Product Canvas.



10. Don't solely rely on user stories

Creating a great user experience (UX) requires more than writing user stories. Also consider the user journeys and interactions, the visual design, and the nonfunctional properties of your product. This results in a holistic description that makes it easier to identify risks and assumptions, and it increases the chances of creating a product with the right user experience.

http://www.romanpichler.com/blog/user-stories/5-common-user-story-mistakes/

5 Common User Story Mistakes

Posted on Wednesday 12th June 2013

Summary

This post discusses five common user story mistakes and helps you overcome them.

User stories are a simple, yet effective way to communicate how a user or customer employs a product. But writing user stories that help a team build great software can be challenging. The following list summarises five common user story mistakes to help you improve your stories.

Story Mania

Some product owners and teams are so fond of user stories that everything is expressed as a story. This either results in some rather odd stories – stories that capture the user interface design, complex user interactions, and technical requirements; or these aspects are simply overlooked.

Like any technique, user story writing has its strengths and limitations. I find stories particularly well suited to capture product functionality, and when applied properly, nonfunctional requirements. But user interface design and complex user interactions are better described by other techniques including design sketches, mock-ups, scenarios, and storyboards. Complement your user stories therefore with other techniques, and don't feel obliged to only use stories.

User Incognito

A user story tells a story about a user interacting with the product. Some stories, however, omit the beneficiary altogether, or they talk about a mysterious user as in "As the user, I want to …" But if it's not clear who the user is, why should the story be developed? How can we be confident that the story will benefit someone?

Make sure that all your stories have a clear user or customer. As I like to work with personas, I use personas in my stories (instead of user roles). This connects each story to the right persona, and it allows me to understand if and to which extend the story addresses the persona's need or goal. To achieve this, I use the following template: As <persona>, I want <something> so that <benefit>.

Disastrous Details

The devil is in the details: Some stories are too big and vague for the team to understand and implement. Others contain too much detail, or even prescribe a solution. Getting the details right seems to be a battle the product owner can only loose.

My recommendation is: Start with big, coarse-grained stories called epics particularly for all new features. Epics allow you to capture an idea without committing to the details. Then break an epic into more detailed user stories. The new user stories replace the epic, and they provide more information about the product's functionality. Pay particular attention to the user stories that are pulled into a sprint. These stories have to be ready: clear, feasible, and testable. (You can learn more about decomposing epics into ready stories in my post "Epics and Ready Stories".)

Progressively refining your stories keeps your Product Canvas or backlog concise, it makes it easier to integrate new insights, and it reduces the effort required to stock your initial canvas or backlog. This is particularly valuable for new products and new features. Make sure you do not prescribe a solution in your stories. Rather focus on the "what", nor the "how". The latter is best captured in an architecture diagram.

Story Handoff

Some product owners diligently write user stories, and give them to the development team in the sprint planning meeting. This handoff is usually suboptimal, as it wastes the team's ideas and knowledge. Stories can hence be inappropriate, difficult to understand, unfeasible and not testable.

User stories, however, are not meant to be standalone documents. They should be complemented by a conversation between the product owner and the team, or even better: written collaboratively. A story wants to capture the essentials, and not specify every detail: The latter would be difficult for new features and too slow and too expensive in an agile / lean context.

User story writing should be a team effort, where product owner and development team create the stories together. Allocate time for a collaborative Product Canvas workshop or backlog grooming meeting, particularly when you develop a new product or new features. Trust me: Better stories and a better product will be your reward.

Criteria Crisis

Acceptance criteria are maybe the most misunderstood part of users stories. I have seen detailed stories with no acceptance criteria, criteria that restate the narrative, and criteria that hide new stories, or even contain entire workflows. The last two mistakes are exemplified by the following story (the narrative is on the card on the left, the "acceptance criteria" on the right):

The idea behind acceptance criteria is simple: They allow you to describe the conditions that have to be fulfilled so that the story is done, that it can be exposed to the users and the other stakeholders. This ensures that you gather feedback and/or release features, and it helps the team plan and track their work: The criteria enrich the story and make it more precise and testable. (The criteria all stories have to fulfil such as "online help is available" are not stated in the acceptance criteria but in the Definition of Done.)

As a rule of thumb, I like to work with three to five criteria per story, and I am not worried if my epics don't have acceptance criteria to start with. Ready stories, however, must provide meaningful criteria. You can find sample acceptance criteria in my posts "Epics and Ready Stories" and "Nonfunctional Requirements".

http://www.romanpichler.com/blog/user-stories/writing-good-user-stories/

10 Tips for Writing Good User Stories

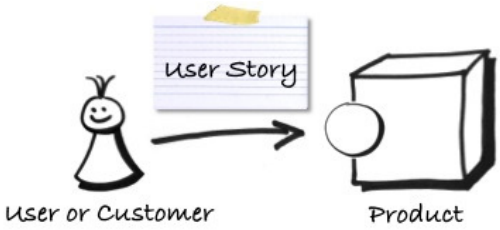
Posted on Thursday 7th October 2010

Summary

User stories are probably the most popular agile technique to capture product functionality: Working with user stories is easy. But writing good stories can be hard. The following ten tips help you create good stories.

1. Focus on the user

As its name suggests, a user story tells a story about a customer or user employing the product. Write therefore stories from the user's perspective and show how a user or customer uses some product functionality.

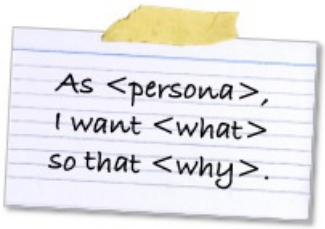


2. Write stories collaboratively

A user story is not a specification, but an communication and collaboration tool. Stories should not be handed off to the development team but be complemented by a conversation: Product owner and team should talk about stories, or even better, write them together: Invite the team to create and detail the stories together. This leverages the creativity and the knowledge of the team and usually results in better user stories.

3. Keep your stories simple and concise

Write your stories so that they are easy to understand. Keep them simple and concise. Avoid confusing and ambiguous terms, and use active voice. Focus on what's important, and leave out non-essential information. The following template puts the user or customer into the story and makes its benefit explicit. But use the template only if it is helpful. Try out different ways to write your stories to understand what works best for you.

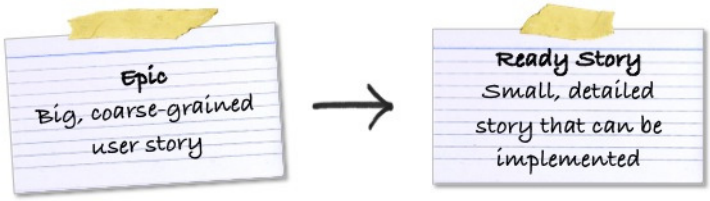


4. Start with epics

Epics are big, coarse-grained user stories. Starting with epics allows you to sketch the product functionality without committing to the details. This is particularly helpful for new products and new features, as it buys you time to learn more about the users and how to best meet their needs. It also reduces the time and effort required to integrate new insights: If you have lots of detailed stories, then it's often tricky to relate the feedback you receive to the right stories. I like to use the personas with their goals to discover the right epics.

5. Decompose your stories until they are ready

Break your epics into smaller, detailed stories until they are ready: clear, feasible, and testable. Everyone should have a shared understanding of the story's meaning; the story should not too big, and there has to be an effective way to determine if the story is done.



6. Add acceptance criteria

As you decompose epics into smaller stories, remember to add acceptance criteria. Acceptance criteria complement the story's narrative: They allow you to describe the conditions that have to be fulfilled so that the story is done. The criteria enrich the story and make it more precise and testable, and they ensures that the story can be demoed or released to the users and the other stakeholders.

7. Group user stories into themes

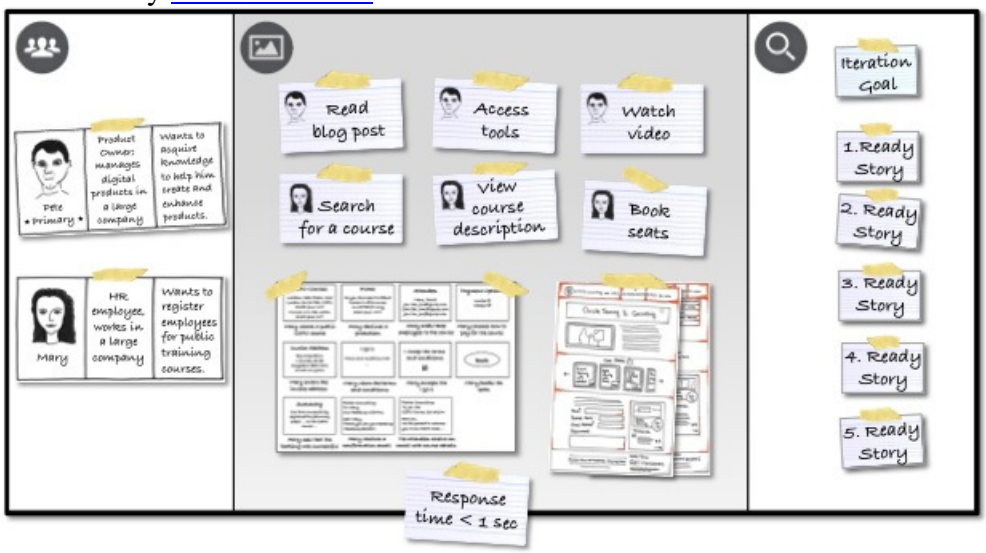
Using themes helps you organise your stories. Each theme is a group of related stories. Sample themes for a mobile phone are email, calendar, voice communication, and organizer, for instance. Themes make it easier to check for completeness and consistency, they structure your product canvas or product backlog, and they facilitate prioritisation.

8. Use paper cards

Paper cards are not only cheap and easy to use. They facilitate collaboration, as everyone can take a card and jot down an idea. Cards can also be easily grouped on the table or wall to check for consistency and completeness. Even if your stories are stored electronically, it is worthwhile to use paper cards when you write new stories.

9. Keep your stories visible and accessible

Stories want to communication information. Don't hide them on a network drive, the corporate intranet jungle, or a licensed tool. Make them visible instead, for instance, by putting them up on the wall. A great tool to discover, visualise, and manage your stories is my Product Canvas.



10. Don't solely rely on user stories

Creating a great user experience (UX) requires more than writing user stories. Also consider the user journeys and interactions, the visual design, and the nonfunctional properties of your product. This results in a holistic description that makes it easier to identify risks and assumptions, and it increases the chances of creating a product with the right user experience.

http://www.romanpichler.com/blog/user-stories/5-common-user-story-mistakes/

5 Common User Story Mistakes

Posted on Wednesday 12th June 2013

Summary

This post discusses five common user story mistakes and helps you overcome them.

User stories are a simple, yet effective way to communicate how a user or customer employs a product. But writing user stories that help a team build great software can be challenging. The following list summarises five common user story mistakes to help you improve your stories.

Story Mania

Some product owners and teams are so fond of user stories that everything is expressed as a story. This either results in some rather odd stories – stories that capture the user interface design, complex user interactions, and technical requirements; or these aspects are simply overlooked.

Like any technique, user story writing has its strengths and limitations. I find stories particularly well suited to capture product functionality, and when applied properly, nonfunctional requirements. But user interface design and complex user interactions are better described by other techniques including design sketches, mock-ups, scenarios, and storyboards. Complement your user stories therefore with other techniques, and don't feel obliged to only use stories.

User Incognito

A user story tells a story about a user interacting with the product. Some stories, however, omit the beneficiary altogether, or they talk about a mysterious user as in "As the user, I want to …" But if it's not clear who the user is, why should the story be developed? How can we be confident that the story will benefit someone?

Make sure that all your stories have a clear user or customer. As I like to work with personas, I use personas in my stories (instead of user roles). This connects each story to the right persona, and it allows me to understand if and to which extend the story addresses the persona's need or goal. To achieve this, I use the following template: As <persona>, I want <something> so that <benefit>.

Disastrous Details

The devil is in the details: Some stories are too big and vague for the team to understand and implement. Others contain too much detail, or even prescribe a solution. Getting the details right seems to be a battle the product owner can only loose.

My recommendation is: Start with big, coarse-grained stories called epics particularly for all new features. Epics allow you to capture an idea without committing to the details. Then break an epic into more detailed user stories. The new user stories replace the epic, and they provide more information about the product's functionality. Pay particular attention to the user stories that are pulled into a sprint. These stories have to be ready: clear, feasible, and testable. (You can learn more about decomposing epics into ready stories in my post "Epics and Ready Stories".)

Progressively refining your stories keeps your Product Canvas or backlog concise, it makes it easier to integrate new insights, and it reduces the effort required to stock your initial canvas or backlog. This is particularly valuable for new products and new features. Make sure you do not prescribe a solution in your stories. Rather focus on the "what", nor the "how". The latter is best captured in an architecture diagram.

Story Handoff

Some product owners diligently write user stories, and give them to the development team in the sprint planning meeting. This handoff is usually suboptimal, as it wastes the team's ideas and knowledge. Stories can hence be inappropriate, difficult to understand, unfeasible and not testable.

User stories, however, are not meant to be standalone documents. They should be complemented by a conversation between the product owner and the team, or even better: written collaboratively. A story wants to capture the essentials, and not specify every detail: The latter would be difficult for new features and too slow and too expensive in an agile / lean context.

User story writing should be a team effort, where product owner and development team create the stories together. Allocate time for a collaborative Product Canvas workshop or backlog grooming meeting, particularly when you develop a new product or new features. Trust me: Better stories and a better product will be your reward.

Criteria Crisis

Acceptance criteria are maybe the most misunderstood part of users stories. I have seen detailed stories with no acceptance criteria, criteria that restate the narrative, and criteria that hide new stories, or even contain entire workflows. The last two mistakes are exemplified by the following story (the narrative is on the card on the left, the "acceptance criteria" on the right):

The idea behind acceptance criteria is simple: They allow you to describe the conditions that have to be fulfilled so that the story is done, that it can be exposed to the users and the other stakeholders. This ensures that you gather feedback and/or release features, and it helps the team plan and track their work: The criteria enrich the story and make it more precise and testable. (The criteria all stories have to fulfil such as "online help is available" are not stated in the acceptance criteria but in the Definition of Done.)

As a rule of thumb, I like to work with three to five criteria per story, and I am not worried if my epics don't have acceptance criteria to start with. Ready stories, however, must provide meaningful criteria. You can find sample acceptance criteria in my posts "Epics and Ready Stories" and "Nonfunctional Requirements".

http://www.romanpichler.com/blog/user-stories/writing-good-user-stories/

10 Tips for Writing Good User Stories

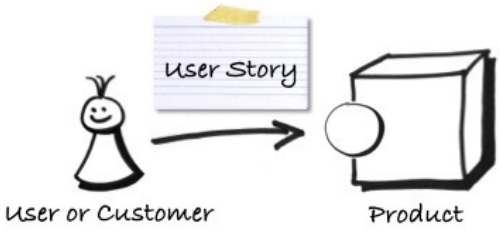
Posted on Thursday 7th October 2010

Summary

User stories are probably the most popular agile technique to capture product functionality: Working with user stories is easy. But writing good stories can be hard. The following ten tips help you create good stories.

1. Focus on the user

As its name suggests, a user story tells a story about a customer or user employing the product. Write therefore stories from the user's perspective and show how a user or customer uses some product functionality.

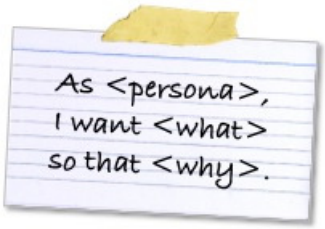


2. Write stories collaboratively

A user story is not a specification, but an communication and collaboration tool. Stories should not be handed off to the development team but be complemented by a conversation: Product owner and team should talk about stories, or even better, write them together: Invite the team to create and detail the stories together. This leverages the creativity and the knowledge of the team and usually results in better user stories.

3. Keep your stories simple and concise

Write your stories so that they are easy to understand. Keep them simple and concise. Avoid confusing and ambiguous terms, and use active voice. Focus on what's important, and leave out non-essential information. The following template puts the user or customer into the story and makes its benefit explicit. But use the template only if it is helpful. Try out different ways to write your stories to understand what works best for you.

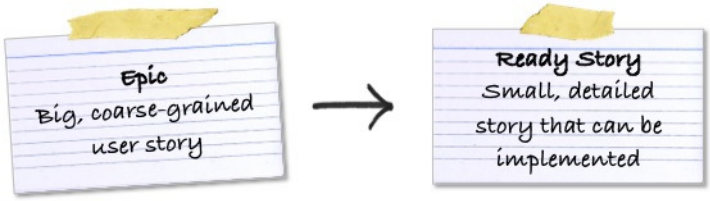


4. Start with epics

Epics are big, coarse-grained user stories. Starting with epics allows you to sketch the product functionality without committing to the details. This is particularly helpful for new products and new features, as it buys you time to learn more about the users and how to best meet their needs. It also reduces the time and effort required to integrate new insights: If you have lots of detailed stories, then it's often tricky to relate the feedback you receive to the right stories. I like to use the personas with their goals to discover the right epics.

5. Decompose your stories until they are ready

Break your epics into smaller, detailed stories until they are ready: clear, feasible, and testable. Everyone should have a shared understanding of the story's meaning; the story should not too big, and there has to be an effective way to determine if the story is done.



6. Add acceptance criteria

As you decompose epics into smaller stories, remember to add acceptance criteria. Acceptance criteria complement the story's narrative: They allow you to describe the conditions that have to be fulfilled so that the story is done. The criteria enrich the story and make it more precise and testable, and they ensures that the story can be demoed or released to the users and the other stakeholders.

7. Group user stories into themes

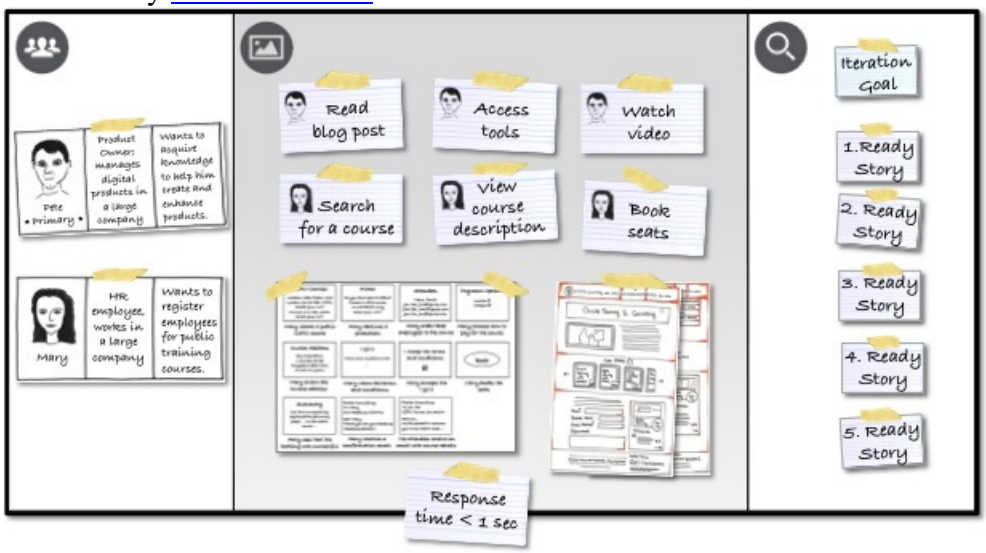
Using themes helps you organise your stories. Each theme is a group of related stories. Sample themes for a mobile phone are email, calendar, voice communication, and organizer, for instance. Themes make it easier to check for completeness and consistency, they structure your product canvas or product backlog, and they facilitate prioritisation.

8. Use paper cards

Paper cards are not only cheap and easy to use. They facilitate collaboration, as everyone can take a card and jot down an idea. Cards can also be easily grouped on the table or wall to check for consistency and completeness. Even if your stories are stored electronically, it is worthwhile to use paper cards when you write new stories.

9. Keep your stories visible and accessible

Stories want to communication information. Don't hide them on a network drive, the corporate intranet jungle, or a licensed tool. Make them visible instead, for instance, by putting them up on the wall. A great tool to discover, visualise, and manage your stories is my Product Canvas.



10. Don't solely rely on user stories

Creating a great user experience (UX) requires more than writing user stories. Also consider the user journeys and interactions, the visual design, and the nonfunctional properties of your product. This results in a holistic description that makes it easier to identify risks and assumptions, and it increases the chances of creating a product with the right user experience.

http://www.romanpichler.com/blog/user-stories/5-common-user-story-mistakes/

5 Common User Story Mistakes

Posted on Wednesday 12th June 2013

Summary

This post discusses five common user story mistakes and helps you overcome them.

User stories are a simple, yet effective way to communicate how a user or customer employs a product. But writing user stories that help a team build great software can be challenging. The following list summarises five common user story mistakes to help you improve your stories.

Story Mania

Some product owners and teams are so fond of user stories that everything is expressed as a story. This either results in some rather odd stories – stories that capture the user interface design, complex user interactions, and technical requirements; or these aspects are simply overlooked.

Like any technique, user story writing has its strengths and limitations. I find stories particularly well suited to capture product functionality, and when applied properly, nonfunctional requirements. But user interface design and complex user interactions are better described by other techniques including design sketches, mock-ups, scenarios, and storyboards. Complement your user stories therefore with other techniques, and don't feel obliged to only use stories.

User Incognito

A user story tells a story about a user interacting with the product. Some stories, however, omit the beneficiary altogether, or they talk about a mysterious user as in "As the user, I want to …" But if it's not clear who the user is, why should the story be developed? How can we be confident that the story will benefit someone?

Make sure that all your stories have a clear user or customer. As I like to work with personas, I use personas in my stories (instead of user roles). This connects each story to the right persona, and it allows me to understand if and to which extend the story addresses the persona's need or goal. To achieve this, I use the following template: As <persona>, I want <something> so that <benefit>.

Disastrous Details

The devil is in the details: Some stories are too big and vague for the team to understand and implement. Others contain too much detail, or even prescribe a solution. Getting the details right seems to be a battle the product owner can only loose.

My recommendation is: Start with big, coarse-grained stories called epics particularly for all new features. Epics allow you to capture an idea without committing to the details. Then break an epic into more detailed user stories. The new user stories replace the epic, and they provide more information about the product's functionality. Pay particular attention to the user stories that are pulled into a sprint. These stories have to be ready: clear, feasible, and testable. (You can learn more about decomposing epics into ready stories in my post "Epics and Ready Stories".)

Progressively refining your stories keeps your Product Canvas or backlog concise, it makes it easier to integrate new insights, and it reduces the effort required to stock your initial canvas or backlog. This is particularly valuable for new products and new features. Make sure you do not prescribe a solution in your stories. Rather focus on the "what", nor the "how". The latter is best captured in an architecture diagram.

Story Handoff

Some product owners diligently write user stories, and give them to the development team in the sprint planning meeting. This handoff is usually suboptimal, as it wastes the team's ideas and knowledge. Stories can hence be inappropriate, difficult to understand, unfeasible and not testable.

User stories, however, are not meant to be standalone documents. They should be complemented by a conversation between the product owner and the team, or even better: written collaboratively. A story wants to capture the essentials, and not specify every detail: The latter would be difficult for new features and too slow and too expensive in an agile / lean context.

User story writing should be a team effort, where product owner and development team create the stories together. Allocate time for a collaborative Product Canvas workshop or backlog grooming meeting, particularly when you develop a new product or new features. Trust me: Better stories and a better product will be your reward.

Criteria Crisis

Acceptance criteria are maybe the most misunderstood part of users stories. I have seen detailed stories with no acceptance criteria, criteria that restate the narrative, and criteria that hide new stories, or even contain entire workflows. The last two mistakes are exemplified by the following story (the narrative is on the card on the left, the "acceptance criteria" on the right):

The idea behind acceptance criteria is simple: They allow you to describe the conditions that have to be fulfilled so that the story is done, that it can be exposed to the users and the other stakeholders. This ensures that you gather feedback and/or release features, and it helps the team plan and track their work: The criteria enrich the story and make it more precise and testable. (The criteria all stories have to fulfil such as "online help is available" are not stated in the acceptance criteria but in the Definition of Done.)

As a rule of thumb, I like to work with three to five criteria per story, and I am not worried if my epics don't have acceptance criteria to start with. Ready stories, however, must provide meaningful criteria. You can find sample acceptance criteria in my posts "Epics and Ready Stories" and "Nonfunctional Requirements".

http://www.romanpichler.com/blog/user-stories/writing-good-user-stories/

10 Tips for Writing Good User Stories

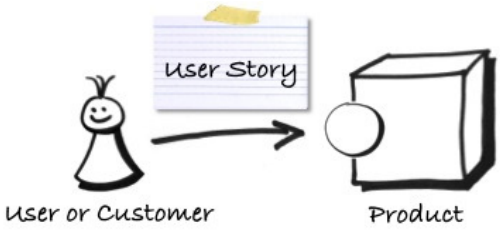
Posted on Thursday 7th October 2010

Summary

User stories are probably the most popular agile technique to capture product functionality: Working with user stories is easy. But writing good stories can be hard. The following ten tips help you create good stories.

1. Focus on the user

As its name suggests, a user story tells a story about a customer or user employing the product. Write therefore stories from the user's perspective and show how a user or customer uses some product functionality.

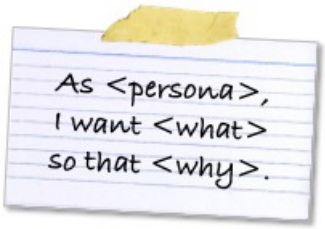


2. Write stories collaboratively

A user story is not a specification, but an communication and collaboration tool. Stories should not be handed off to the development team but be complemented by a conversation: Product owner and team should talk about stories, or even better, write them together: Invite the team to create and detail the stories together. This leverages the creativity and the knowledge of the team and usually results in better user stories.

3. Keep your stories simple and concise

Write your stories so that they are easy to understand. Keep them simple and concise. Avoid confusing and ambiguous terms, and use active voice. Focus on what's important, and leave out non-essential information. The following template puts the user or customer into the story and makes its benefit explicit. But use the template only if it is helpful. Try out different ways to write your stories to understand what works best for you.

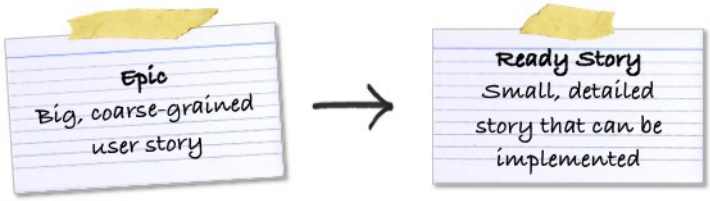


4. Start with epics

Epics are big, coarse-grained user stories. Starting with epics allows you to sketch the product functionality without committing to the details. This is particularly helpful for new products and new features, as it buys you time to learn more about the users and how to best meet their needs. It also reduces the time and effort required to integrate new insights: If you have lots of detailed stories, then it's often tricky to relate the feedback you receive to the right stories. I like to use the personas with their goals to discover the right epics.

5. Decompose your stories until they are ready

Break your epics into smaller, detailed stories until they are ready: clear, feasible, and testable. Everyone should have a shared understanding of the story's meaning; the story should not too big, and there has to be an effective way to determine if the story is done.



6. Add acceptance criteria

As you decompose epics into smaller stories, remember to add acceptance criteria. Acceptance criteria complement the story's narrative: They allow you to describe the conditions that have to be fulfilled so that the story is done. The criteria enrich the story and make it more precise and testable, and they ensures that the story can be demoed or released to the users and the other stakeholders.

7. Group user stories into themes

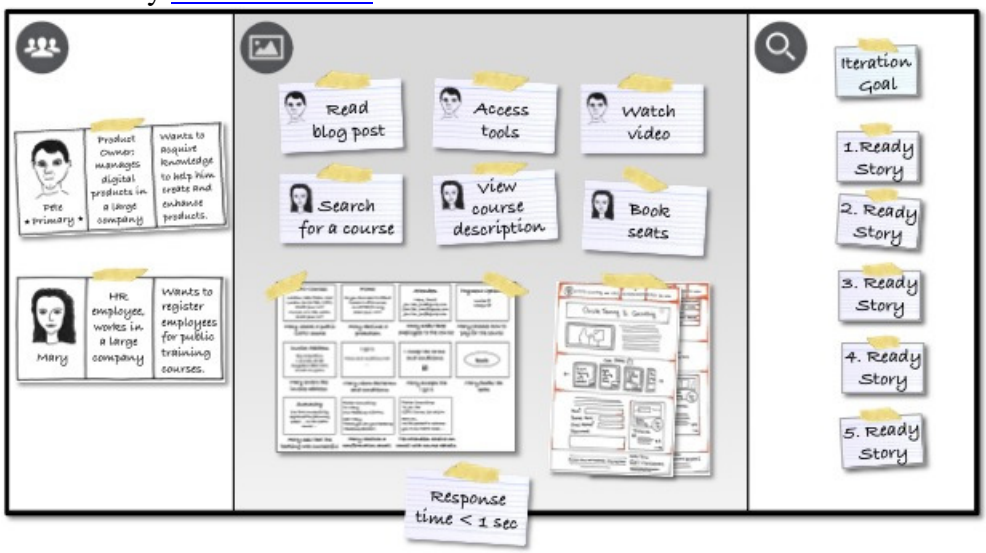
Using themes helps you organise your stories. Each theme is a group of related stories. Sample themes for a mobile phone are email, calendar, voice communication, and organizer, for instance. Themes make it easier to check for completeness and consistency, they structure your product canvas or product backlog, and they facilitate prioritisation.

8. Use paper cards

Paper cards are not only cheap and easy to use. They facilitate collaboration, as everyone can take a card and jot down an idea. Cards can also be easily grouped on the table or wall to check for consistency and completeness. Even if your stories are stored electronically, it is worthwhile to use paper cards when you write new stories.

9. Keep your stories visible and accessible

Stories want to communication information. Don't hide them on a network drive, the corporate intranet jungle, or a licensed tool. Make them visible instead, for instance, by putting them up on the wall. A great tool to discover, visualise, and manage your stories is my Product Canvas.



10. Don't solely rely on user stories

Creating a great user experience (UX) requires more than writing user stories. Also consider the user journeys and interactions, the visual design, and the nonfunctional properties of your product. This results in a holistic description that makes it easier to identify risks and assumptions, and it increases the chances of creating a product with the right user experience.

http://www.romanpichler.com/blog/user-stories/5-common-user-story-mistakes/

5 Common User Story Mistakes

Posted on Wednesday 12th June 2013

Summary

This post discusses five common user story mistakes and helps you overcome them.

User stories are a simple, yet effective way to communicate how a user or customer employs a product. But writing user stories that help a team build great software can be challenging. The following list summarises five common user story mistakes to help you improve your stories.

Story Mania

Some product owners and teams are so fond of user stories that everything is expressed as a story. This either results in some rather odd stories – stories that capture the user interface design, complex user interactions, and technical requirements; or these aspects are simply overlooked.

Like any technique, user story writing has its strengths and limitations. I find stories particularly well suited to capture product functionality, and when applied properly, nonfunctional requirements. But user interface design and complex user interactions are better described by other techniques including design sketches, mock-ups, scenarios, and storyboards. Complement your user stories therefore with other techniques, and don't feel obliged to only use stories.

User Incognito

A user story tells a story about a user interacting with the product. Some stories, however, omit the beneficiary altogether, or they talk about a mysterious user as in "As the user, I want to …" But if it's not clear who the user is, why should the story be developed? How can we be confident that the story will benefit someone?

Make sure that all your stories have a clear user or customer. As I like to work with personas, I use personas in my stories (instead of user roles). This connects each story to the right persona, and it allows me to understand if and to which extend the story addresses the persona's need or goal. To achieve this, I use the following template: As <persona>, I want <something> so that <benefit>.

Disastrous Details

The devil is in the details: Some stories are too big and vague for the team to understand and implement. Others contain too much detail, or even prescribe a solution. Getting the details right seems to be a battle the product owner can only loose.

My recommendation is: Start with big, coarse-grained stories called epics particularly for all new features. Epics allow you to capture an idea without committing to the details. Then break an epic into more detailed user stories. The new user stories replace the epic, and they provide more information about the product's functionality. Pay particular attention to the user stories that are pulled into a sprint. These stories have to be ready: clear, feasible, and testable. (You can learn more about decomposing epics into ready stories in my post "Epics and Ready Stories".)

Progressively refining your stories keeps your Product Canvas or backlog concise, it makes it easier to integrate new insights, and it reduces the effort required to stock your initial canvas or backlog. This is particularly valuable for new products and new features. Make sure you do not prescribe a solution in your stories. Rather focus on the "what", nor the "how". The latter is best captured in an architecture diagram.

Story Handoff

Some product owners diligently write user stories, and give them to the development team in the sprint planning meeting. This handoff is usually suboptimal, as it wastes the team's ideas and knowledge. Stories can hence be inappropriate, difficult to understand, unfeasible and not testable.

User stories, however, are not meant to be standalone documents. They should be complemented by a conversation between the product owner and the team, or even better: written collaboratively. A story wants to capture the essentials, and not specify every detail: The latter would be difficult for new features and too slow and too expensive in an agile / lean context.

User story writing should be a team effort, where product owner and development team create the stories together. Allocate time for a collaborative Product Canvas workshop or backlog grooming meeting, particularly when you develop a new product or new features. Trust me: Better stories and a better product will be your reward.

Criteria Crisis

Acceptance criteria are maybe the most misunderstood part of users stories. I have seen detailed stories with no acceptance criteria, criteria that restate the narrative, and criteria that hide new stories, or even contain entire workflows. The last two mistakes are exemplified by the following story (the narrative is on the card on the left, the "acceptance criteria" on the right):

The idea behind acceptance criteria is simple: They allow you to describe the conditions that have to be fulfilled so that the story is done, that it can be exposed to the users and the other stakeholders. This ensures that you gather feedback and/or release features, and it helps the team plan and track their work: The criteria enrich the story and make it more precise and testable. (The criteria all stories have to fulfil such as "online help is available" are not stated in the acceptance criteria but in the Definition of Done.)

As a rule of thumb, I like to work with three to five criteria per story, and I am not worried if my epics don't have acceptance criteria to start with. Ready stories, however, must provide meaningful criteria. You can find sample acceptance criteria in my posts "Epics and Ready Stories" and "Nonfunctional Requirements".