JUST PLAIN Agile
Train. Coach. Transform. Perform.

User stories

User stories define the behaviour of the system and the resultant business value to the user. It is a "*brief statement of intent that describes something the system needs to do for the user*". [Leffingwell] A story does not specify all the detail, but is a **reminder to have a conversation** about the detail. Ideally stories are written by users themselves (hence the name!) and always phrased in a way that a business user is very clear on what they mean. A business user therefore can prioritise a story, explain what is required for the story and come up with acceptance criteria for the story. The stories belong to them.

User stories are the agile replacement for software requirements documents. Yet they are not requirements as such, since they always describe the delta between what is and what should be, rather than being a detailed description of a feature. They are informal, ideally written up on index cards as they are formulated and discarded after their implementation. If there is a need to document anything around the feature the story pertains to, eg. business rules, the user experience, a technical aspect, etc. this should still be done by the technical team, but with a light touch. It is the software that is valuable, much more so than the documentation about the software.

The most common form of the story is:

> *As a* <role>*, I can* <activity> *so that* <business value>

Eg.

> *As a* helpdesk operator*, I can* search for a client *so that* I can retrieve a client's information quickly when somebody calls in.

They are often captured on index cards, and the Card/Conversation/Confirmation format (Ron Jeffries) refers to the practice to record the acceptance criteria on the back of the card.

The **INVEST** principles for good stories [Bill Wake www.XP123.org

I ndependent – developed, tested, delivered on its own. Has independent value.

N egotiable – no contract, but placeholder to discuss, develop, test, accept a requirement

V aluable – most important principle. Challenging. Even tech stories must have value

E stimable – As a team. If you can't, break it up. If you don't know, spike it.

S mall – must fit into 1 iteration. Do one thing, keep it small, make it smaller still

T estable – don't get into an iteration if they can't get out

Just Plain Agile Consulting | CK 2006/116716/23 | VAT Registration number: 4910261892
P. O. Box 6973, Welgemoed, 7538 | +27-61-835-6809 | info@justplainagile.co.za

JUST PLAIN Agile
Train. Coach. Transform. Perform.

Splitting stories

One of the biggest changes and challenges when a team starts developing in an agile way is how to break things up to do them in small chunks that still deliver value incrementally. At the heart of this challenge is the ability to break user stories up in meaningful smaller stories. Below are some of the techniques that can help.

Acceptance criteria The easiest way to split a story is by using the acceptance scenarios as a guideline.	*"...I can enter a valid new password.."* can be split into: • A story that checks that the new password is of a certain length • a story that adds the rules around upper and lower case, digits and special characters • a story that checks that the password has not been used before
Business rule validation Similar to the technique above, start with a sunny day scenario, i.e. everything is valid, then add exception scenarios	*"... I can register as a user of the site..." can be split into:* • a story that allows the user basic registration, only checking that the username is not already in use • a story that checks that the username is a valid email address by sending a mail • a story that checks that the password conforms to the password policy
Sophistication Start with a simplistic implementation, then add sophistication. This can be in terms of the user interface, the underlying functionality, the integration with external systems and so forth.	*"... I can select a username from a valid set of suggestions..."* can be split into: • A story that asks for a username and gives an error if it has been used before • a story that suggests alternatives when the username selected has been used before by adding numerical digits • a story that suggests a username from the start
Variances in data Cope with a basic data set, then add more	*"... I can register using my details from a social media site.."* can be split into • a story to register using data from Facebook • a story to register using data from LinkedIn • etc.
CRUD instead of Maintain/Manage Specify stories for Create, Read, Update and Delete seperately	*"... I can maintain my logon.."* can be split into • a story to register (Create the logon) • a story to change the password (Update) • a story to lookup a forgotten username (Read) • a story to deregister or expire a logon (Delete)
Major effort Break down major underlying technical details into multiple stories	*"... I can pay for my membership.."* can be split into • a story that allows payment through EFT • a story that allows payment through Paypal • a story that allows debit card payments through VISA Electron • a story that allows credit card payments through Mastercard, etc, etc.
User device variations Choose one device, then add device options	*"... I can register for membership on any device.."* can be split into • a story to register using a web page • a story to register using the browser on a phone • a story to register on a smart phone using an app • a story to register using a feature phone
Role differentiations Similar to *Sophistication* above, choose a basic role, then add functionality for more complex roles	*"... I can register as a user of the site.."* can be split into • a story for an internal user • a story for a call centre user • a story for a known client • a story for a prospect

Just Plain Agile Consulting | CK 2006/116716/23 | VAT Registration number: 4910261892
P. O. Box 6973, Welgemoed, 7538 | +27-61-835-6809 | info@justplainagile.co.za