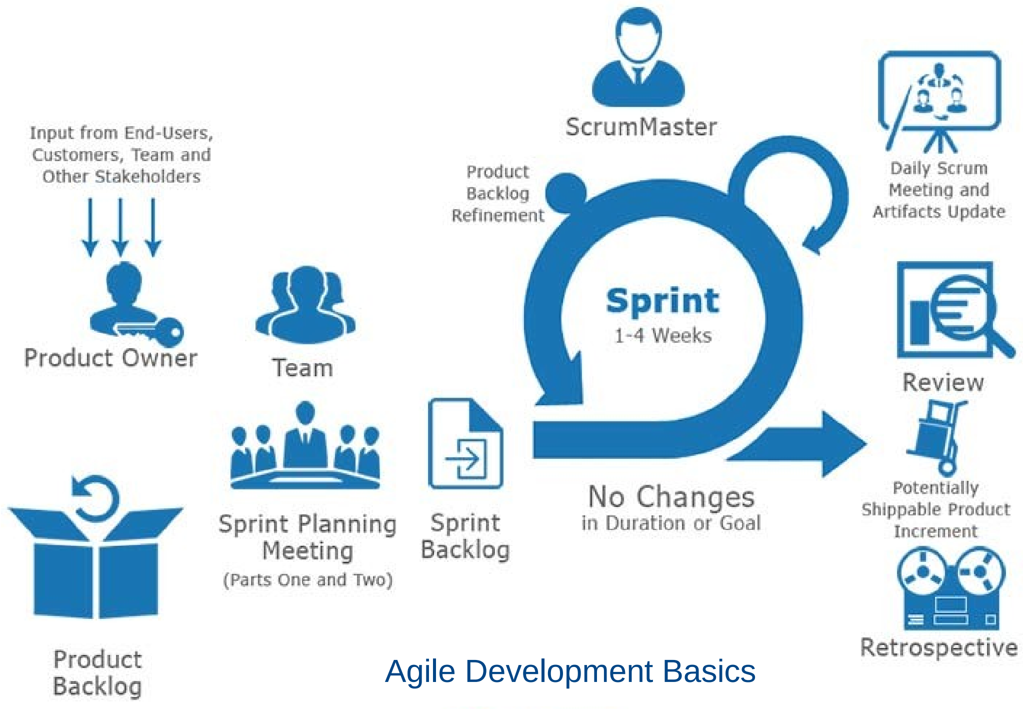
Write a great BI user story

Pentaho Agile BI Methodology

This document describes how to write a good user story to be part of the product backlog in a typical agile development process as shown below. The user stories are what make up the initial "product backlog" and supposed to be sufficiently fine grained to be separately taken into "sprint backlogs" for incremental delivery of the final product. User stories do not change in scope during the sprints, however they continue to get updated and refined with additional insights during the lifetime of the project.


Input from End-Users, Customers, Team and Other Stakeholders
Product Owner
Team
ScrumMaster
Product Backlog Refinement
Sprint
1-4 Weeks
Daily Scrum Meeting and Artifacts Update
Review
Potentially Shippable Product Increment
Retrospective
No Changes in Duration or Goal
Sprint Planning Meeting (Parts One and Two)
Sprint Backlog
Product Backlog

Agile Development Basics

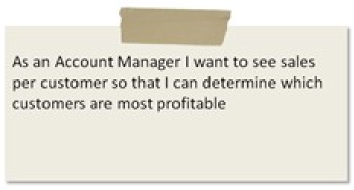
What Is A User Story?

A user story represents a small piece of business value that a team can deliver in an sprint. While traditional requirements gathering (e.g. when using "use cases") tries to be as detailed as possible upfront, a user story is defined incrementally, in three stages:

- The brief description of the need
- The conversations that happen during backlog grooming and iteration planning to solidify the details
- The tests that confirm the story's satisfactory completion

The above is also sometimes described using the three C's of user stories.

- **Card**: User stories are written on cards (or post it notes) called story cards. The card has just enough text to identify the requirement and remember what the user story is. They are used for planning and notes will be added to capture priority and cost/estimation.



As an Account Manager I want to see sales per customer so that I can determine which customers are most profitable

- **Conversation**: The requirement itself is communicated from the business to the data warehouse developer through conversation. This verbal exchange of thoughts, opinions and feelings is often considered the most important part of user stories
- **Confirmation**: This adds the acceptance tests, that document the details that can be used to determine when a story is complete. The business communicates how they will confirm that the story has been delivered

Why User Stories?

Working with user stories in an agile approach is inherently different from detailed upfront use case descriptions. The objectives of working with incrementally developed user stories are the following:

- Keep yourself expressing business value
- Avoid introducing detail too early that would prevent design options
  and inappropriately lock developers into one solution
- Avoid the appearance of false completeness and clarity
- Get to small enough chunks that invite negotiation and movement in the backlog
- Leave the technical functions to the architect, developers, testers, and so on

Criteria for a good user story

Well-formed stories will meet the criteria of Bill Wake's INVEST acronym:

- **Independent**: We want to be able to develop in any sequence.
- **Negotiable**: Avoid too much detail. Flexible so the team can adjust how much of the story to implement.
- **Valuable**: Users or customers get some value from the story.
- **Estimatable**: The team must be able to use them for planning.
- **Small**: Large stories are harder to estimate and plan. By the time of iteration planning, the story should be able to be designed, coded, and tested within the iteration.
- **Testable**: Document acceptance criteria, or the definition of done for the story, which lead to test cases.

How Do I Write User Stories?

When getting started with stories, a template can help ensure that you don't inadvertently start writing technical tasks:

As a <user type>, I want to <function> so that <benefit> .

Examples:

- As a consumer, I want shopping cart functionality to easily purchase items online.
- As an executive, I want to generate a report to understand which departments need to improve their productivity.

Try to avoid the generic role of “User” when writing user stories. User stories are about all of the role who interact with the system or who realize some value or benefit from the system. Not all actors are end users. For example, a role could be another system or someone who wants certain functionality in order to buy your product but will never actually use the product. It may be useful to create aggregate roles (such as consumer) and specialized roles (such as browser or frequent shopper).

What Size Should A User Story Be?

A story should be small enough to be coded and tested within an iteration—ideally just a few days. When a story is too large, it is called an epic. Backlog items tend to start as epics when they are lower priority. For release planning, epics should be broken down into smaller chunks, but not so small that you have moved into detailed design.

Types of User Stories in BI Projects

In BI systems, five types of user stories are distinguished

- Data disclosure stories
- Data presentation stories
- Data augmentation stories
- Data validation stories
- Configuration stories

The following paragraphs describe examples of the different types of user stories. However, bear in mind that most of the time, the different types of stories will be interrelated. The challenge is to separate them out in pieces that have individual value to the user and can be delivered separately.

Data disclosure stories are about extracting data from the source system and making it available in self-service BI enviornment. While feels counter-intuivie, in these stories it is very useful to mention the source system. Most of the times the user has a prevalence for a certain system, that he/she already uses in his/her daily work. Adding this information the user story makes it clear what the user actually expects and can at least help to get the conversation going about what is the best source for this data. The risk with data disclosure stories is to make them to large because a single system can contain a lot of information items (e.g. sales orders, price books, shipping information, …). It is best to keep the stories confined to single information items.

Examples:

- As a sales analyst, I want to be able to analyze revenues on sales transactions from the POS system on a weekly basis so that I can find patterns in sales over time
- As a financial controller, I want to be able to view all the outstanding credit lines from the loans system on a monthly basis so that I can verify the balance of the department.

Data presentation stories describe the presentation of the information in a format that the user can easily understand. In order to use the information, the user needs to be able to draw conclusions and explain the conclusions to others. These can be pivot tables and graphs (typically simple user stories), infographics or outcomes of statistical analyses (possibly complex user stories).

Examples:

- As a sales manager I want to have monthly report, which shows me the sales figures by sales agents, so that I can steer the agents to improve the sales.
- As a marketing manager, I want to know whether customer age has a relationship with the product segments bought, so that I can find smarter ways to market our products
- As a corporate risk manager, I need to have a monthly report on Risk Weighted Assets for my bank, so that I can report to the regulator on our positions.

Data augmentation stories are about creating new information based on already existing information. For these derivation, business rules are used. You can derive profit margin based on costs and revenues. You can drive customer age segment based on his date of birth. And if you have done sufficient analysis, you can derive the chance of customer churn with a calculation model that uses recent transaction behavior, website usage and number of helpdesk calls.

Examples:

- As a call center sales agenda, I want to see the customer age segment, so that I can use the communication style and offers that are most likely to fit the customer's taste.
- As a customer account manager, I want to see the customer churn probability for a customer, so that I can

focus on the clients that are most likely to leave.

Data validation stories describe which rules need to be applied to check whether the data extracted is of sufficient quality for the end user to work with, including any actions needed to get the quality to an appropriate level. Not all actions will be committed automatically but at least the discussion on data quality makes the product owner and end-users aware that data quality issues exist.

Examples:

- As a compliance manager, I want to see whether the complaints report is based on all complaints that are issued during the last month, so that I can send the report to the regulator without breaching the rules of the regulator.
- As a data steward for the loans department, I want to be notified that when mortgages exist that do not have proper customer data attached, so that I can take proper actions to keep risks as low as possible.

Finally, **configuration stories** are about enabling maintenance staff and administrators to keep the configurations of the BI system up to date without having to change the code. This can be used for user authorization and security settings, reference data or hierarchies for which no separate source system exists, and so on. Quite often the user in the user story is not a business user, but a user from the IT department, doing maintenance of the system

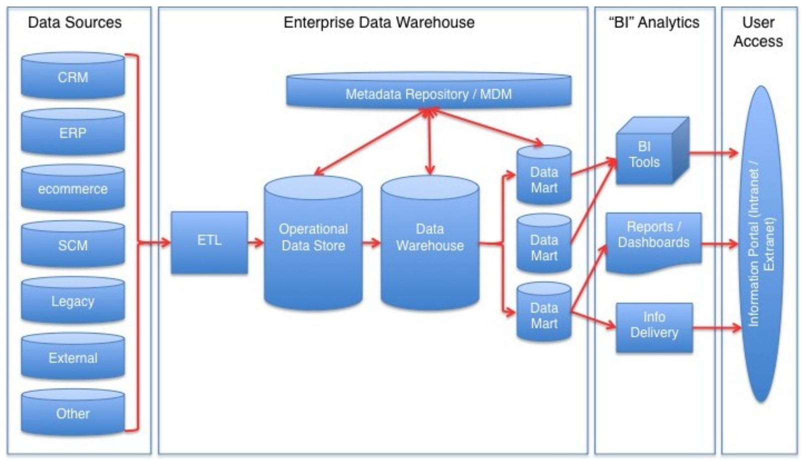
Examples:

- As a product manager I want to be able to configure and update the product hierarchy, so that I can keep the hierarchy up date data for correct analysis
- As a security administrator, I need to be able to configure which users are allowed to see the information for which branches and regions so that I can assure compliance to the data confidentiality rules of the company.

As mentioned above, most of the time, you'll be tempted to merge at least three or four of the above user story types into a single story because the above are all interconnected. One could argue that for any user question you need to go through data extraction, data augmentation/transformation, data validation and data presentation at least. The challenge is to slice the stories so they stay small but still granular enough to individually add value to the business.

Slicing Stories

Typically a data warehouse architecture contains many layers and it is here that we can introduce agility into the process, ensuring that we don't make stories too thick initially and/or can iterate quickly within a user story as we develop it.



There are all these different opportunities as go through each of the layers, to make decisions about how deep into each of those layers we are going to slice. And we must ask ourselves: How can we shrink each step? One example is to do change data capture as a follow on story, instead of in the staging layer. Another example is to do data hygiene, or to pull the data all the way through to the BI layer, without doing data cleansing. You might want to have your business look at the data before deciding whether or not to do data cleansing. If when you pull the data all the way through, and your business sees that only .1% of this field has a data quality issue, they may say "That's fine. That's within tolerable limits. You don't need to ever cleanse that data." You've just saved yourself some work and some time and you can move on to something of more value. Each of these examples are ways you can shrink each step and come back and do it later in the same or in a different story, if that adds value. I would argue that often, with the right toolset, in less than a day data can be extracted, transformed and proposed to the specific end-user, leaving sufficient time to iterate over the requirements and mature the story.

Obviously, you do need to make sure that finally you do adhere to architectural principles set by your lead architect. Slicing the stories thin should not be an excuse to build data marts or cubes directly of your data source, skipping any structured ODS or DWH in the middle tiers.

Conclusion

In an agile development approach a good user story follows the INVEST criteria (Independent, Negotiable, Valuable, Estimatable and Testable), starts off small and through discussion and interaction with the stakeholders and the eventual validation, matures, ensuring alignment with the business needs through this process. Where-as it might seem tough to make a typical BI user story sufficiently thin, because of the various steps to cover in a typical business intelligence architecture, actually many steps can be left out in a first iteration and added back in to complete the story.