User Stories Don't Help Users: Introducing Persona Stories

William Hudson
Syntagm Ltd, Abingdon UK
whudson@acm.org

User stories are one of the most popular alternatives to traditional user requirement specifications (see Figure 1). But despite their promising name, user stories are not about – and don't necessarily help – users at all. In most cases, user stories are written about *roles* that users adopt and take no account of the needs and behaviours of real users. Were that not indictment enough, user stories suffer from demonstrable flaws in structure and are often written by the wrong people at the wrong time.

This article examines the background of user stories in their current form, highlights their failings and proposes a more appropriate alternative for the development of interactive systems; *persona stories*.

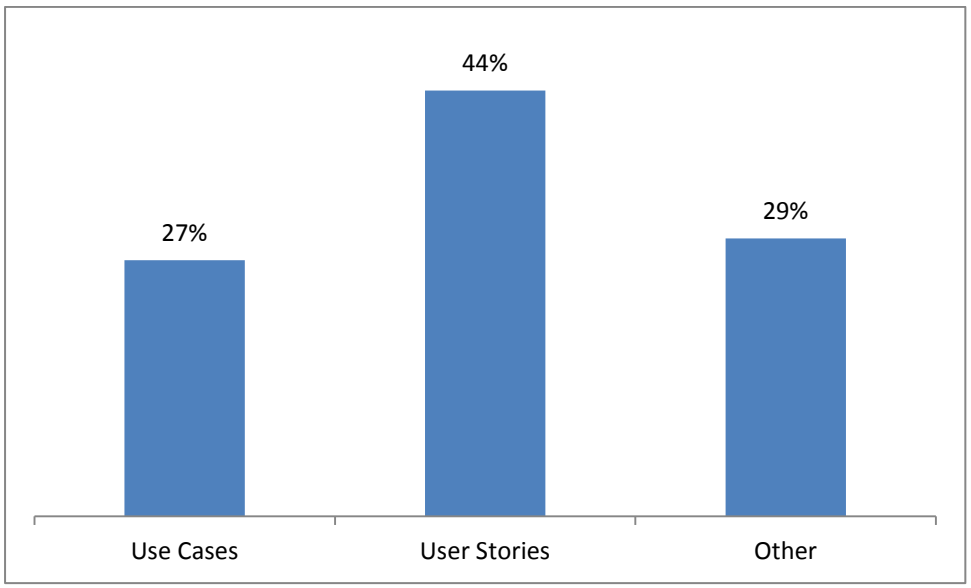

Figure 1, Survey of user requirement methods, April 2013. 'Other' includes some projects where both use cases and user stories were employed (n =112).

1 Background

User stories – as brief scenarios of use written on small cards – were used in the first Extreme Programming project, C3 at Chrysler [1], in 1996 and described by Kent Beck in his book *eXreme Programming Explained* [2] in 1999. The idea of describing requirements as stories of use has a long history, much of it in Human-Computer Interaction. In Scenario-Based Design [3], Jack Carroll traces the thinking about scenarios in design back to a 1959 article by C W Mills [4]. But it was Ivar

Jacobson who, in an early paper on use cases, proposed describing the requirements of industrial systems from this scenario perspective [5].

Like use cases, early user stories did not have a specific form. The lack of form and consistent content was seen as problematic by some in the Agile community and was addressed by a team at Connextra in 2001 [6]. They proposed that user stories should take the form

```
As a <role>, I want <goal/desire> [so that <benefit>]
```

Elements in angled brackets (such as role) are to be supplied and elements in square brackets are optional. This style of user story was popularized by Mike Cohn in *User Stories Applied [7]* in 2004.

2 User Stories

While there is frequent debate on the value of user stories in general and of the Connextra form in particular, both give rise to substantial issues when viewed from an HCI perspective, which I will address in some detail. In particular, the focus on roles is inappropriate for many systems and the narrative structure as outlined above is unsuitable for a number of reasons.

2.1 The Role of Roles

A role is a systemizing concept that has wide-spread use in business and industry. Its primary function is to describe an individual's activities and responsibilities. In software development, Jacobson made use of roles in use cases (here, quoting Kristen Nygaard [8] from a 1986 lecture):

> "A role is defined through a specified task or a group of closely related tasks which are performed by persons during the development and/or operation of a system"

But roles are not as simple as they might first appear. In *Understanding Organizations* [9] Charles Handy devotes an entire chapter to roles and interactions, describing issues such as role ambiguity, role incompatibility, role conflict, role overload, role underload, role strain and many others. The first of these, role ambiguity, is particularly relevant to systems design since it describes the lack of clarity that an individual, their colleagues and the organization itself may have about roles. Those of us conducting user research have usually had first-hand experience of this.

In *The structure of work: Job design and roles* Daniel Ilken and John Hollenbeck describe the issue very explicitly:

> "The simplicity of the role definition as a set of expected behaviours masks the complexity and ambiguity that is discovered as one probes more deeply into the underlying assumptions behind the definition." [10]

Roles are only a tiny part of the picture when it comes to the needs and behaviours of users. They tell us, approximately, what *kind* of activities a user may undertake but they say nothing about *how* and *when* tasks are performed. So, for example, we may identify an accounting or bookkeeping role but we would need to do user research to discover that some tasks are performed many times a day while others are relatively rare. The implications of these differences are very significant for interactive systems design.

Larry Constantine and Lucy Lockwood do acknowledge the need for research and role modelling in their Usage-Cantered Design process [11]. Constantine and Lockwood's user roles are more

descriptive, including characteristic details which result in role names such as SingleTicketPurchaser. However, since this role is somewhat tautologous – anyone purchasing a single ticket is by definition a SingleTicketPurchaser – it is hard to see where the role ends and the use case or user story begins. In this instance, applying the Connextra/Cohn form of user story yields

> As a customer I want to purchase a single ticket so that I can travel

But herein lays a significant problem. Roles are not particularly useful in consumer-oriented systems. We end up with stories with roles like 'customer', 'visitor' or 'subscriber' which tell nothing about what we call the *contexts of use* [12]. These contexts cover a range of issues beyond simple role descriptions. And for many systems these are things that we should research at an early stage. What we need is an approach to requirements that focuses on those differences in the behaviours and needs of users that will require us to provide substantially different forms of interaction. But first, some more challenges with user stories…

2.2 Wrong People

While user stories are the child of extreme programming (XP), they have since been adopted by other Agile approaches, most notably Scrum [13]. In XP and Scrum user stories should be written by the business, product owner, customer team or user representative depending on where and when you look (it changed between the first and second editions of *Extreme Programming Explained*, for example [2, 7, 14]). But as I argued during the early excitement about user stories, no one person (or even small team) can dictate or represent the needs of users [15]. This is particularly true of users on the Agile team who are almost always chosen for the wrong reasons – after all, you wouldn't select someone who wasn't particularly good at their job to act in this capacity, but they may actually be much more representative of real users.

There is a further complication that needs to be mentioned. From my own research, using methods normally applied to the study of autism, I found that technology-focused men working in IT had significantly reduced empathy. Figure 2 shows the result of the study for men (EQ is empathizing quotient, SQ is systemizing quotient [16]). These results are indicative of something that had been observed decades earlier:

> "Programmers dislike activities involving close personal interaction. They prefer to work with things rather than people." [17]

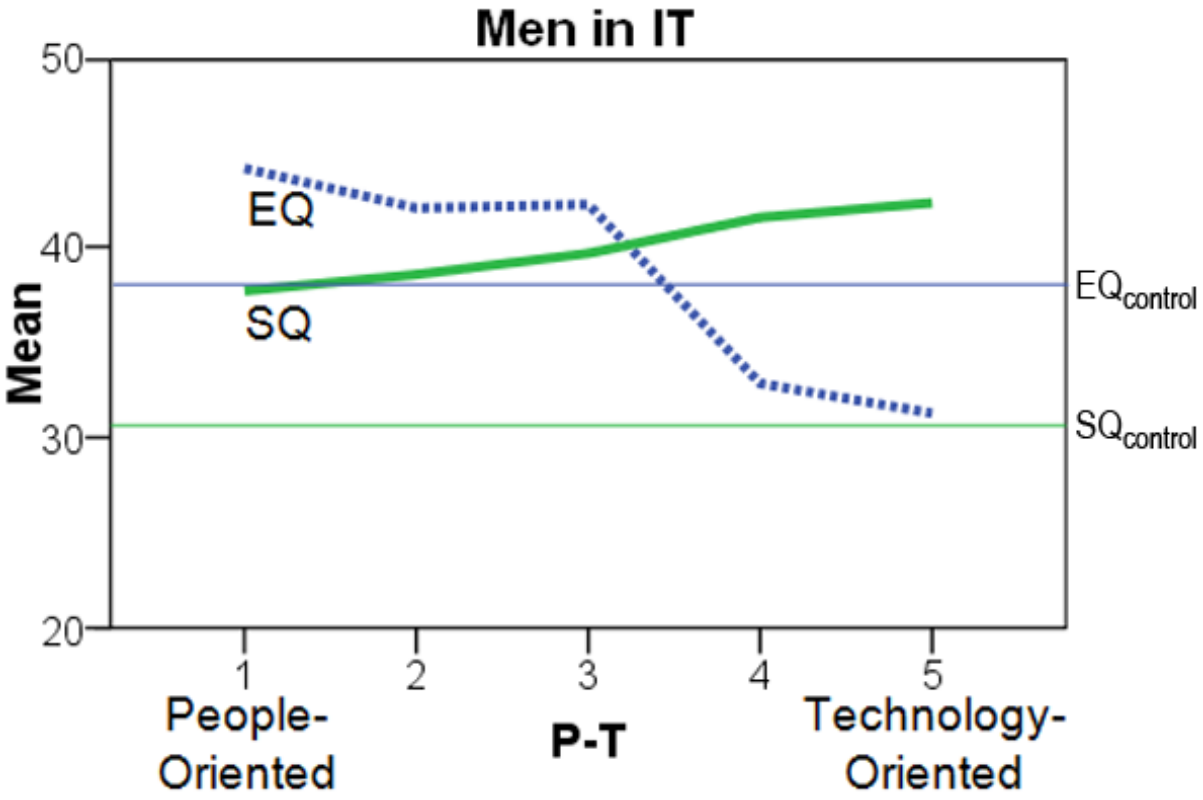

Figure 2, Technology-oriented men showed a marked reduction in empathy (EQ and SQ controls are for the average population). (n=156 men, n=285 women)

Reduced empathy doesn't just explain the lack of enthusiasm for personal interaction, though. It also means that technologists – who are very good at building and understanding systems – find it hard to see a problem from a perspective other than their own. And low empathy isn't confined to technologists; in the normal population empathizing and systemizing skills are unrelated. So we need to take concrete action to ensure that any artefacts describing users' interactions with a system are written by someone with empathizing skills, a good understanding of user behaviour and in such a way as to promote empathy within the team. (Women, on average, have higher empathizing scores than men. Both men and women who were people-oriented in their job roles scored above population averages for empathy in the study.)

2.3 Wrong Time

In XP, user stories are meant to feed into the planning game (adapted from *The New New Product Development Game*[a], a frequently-cited 1986 article in the Harvard Business Review [18]). But if a team is going to make estimates and plans based on user stories they need to at least know what needs to be done and approximately how. This is particularly true of novel systems, where there are no established current practices to be described. That means that we need to give some thought to the purpose, shape and size of the system *before* writing user stories. Unfortunately, many Agile adopters believe that design is a Bad Thing even though this is not actually what the Agile Manifesto and its Twelve Principles of Agile Software say [19]. (Big Design Up Front [20] is bad but rough design

[a] It really is "New New".

is not.) So not surprisingly, estimates and plans made with premature user stories may not be very reliable.

2.4 Structural Flaws

I mentioned at the outset that user stories, certainly in their Connextra/Cohn form are structurally flawed:

1) Roles are not a suitable focus of attention for many systems
2) The 'As a <role> I want…' form is unnecessarily wordy and repetitive
3) The use of the first person is counter-productive

Since I have already outlined the case against roles, let me address the two remaining points. Even in small systems there are likely to be scores if not hundreds of user stories. To read and write 'As a <role> I want…' each and every time is both monotonous and time-consuming. When we look at how people scan and read material, superfluous words at the beginning of sentences are particularly troublesome since they move the more important content further into the text, thereby making it harder to process [21, 22].

On the final point above, there are two separate reasons for declaring that the use of the first person is unhelpful. The first is very familiar to all who have worked in usability and user experience: many usability issues arise because developers assume that the users are similar to themselves. In the majority of cases this is simply untrue although reduced empathy (see section 2.2) makes this hard for developers to understand. In fact a Google search for the phrase "you are not the user" produces over 470,000 results [23].

The second reason is that thinking about yourself is not actually as effective in a design context as thinking about others unknown to you. In four studies published in 2011 researchers found that participants were more creative and better able to solve problems when doing it for distant others (people they did not know personally) rather than close others or themselves [24].

3 Personas and Persona Stories

Given the case against user stories, but the undeniable interest in using something so relatively immediate (compared with traditional requirements or use cases), what alternatives do we have? My belief is that user stories can be adapted to be more user-centred by changing their structure and shifting their focus from roles to 'minimal collaborative personas'.

3.1 Minimal Collaborative Personas

Although the term 'persona' and its related concepts in English are quite old (originating from the Latin for 'person'), Alan Cooper is the first to have applied the term to characterize users of an interactive system during design, in 1999 [25]. Cooper's goal with personas was to give designers focus and to make users seem more like real people. Personas have since become very popular in the fields of usability and user experience but unfortunately the original purpose has become somewhat blurred. It is not uncommon to hear about UX teams spending many weeks or even months developing personas [26]. I have seen individual personas of six pages in length. These are not likely to help give developers focus. Nor is the common practice of the UX team researching and

writing personas in isolation, then providing them as a fait accompli to developers likely to make the personas seem like real people.

To be Agile we need *minimal, collaborative personas*:

> **Minimal:** Each primary persona requires a different user interface. The persona descriptions need to explain what behaviours and needs each persona has that make a different interface necessary. There should be a small amount of back-story (character description) and motivation so that anyone presented with a scenario for our system can read the persona and come away thinking "Ah, that's why we need to do it that way". Ideally, each persona should occupy the front of a sheet of paper. The back of the same sheet should provide practical information along the lines of "how would I know this persona if I saw one?" – used to recruit participants for research and usability evaluation. Specific personal details should be provided to the extent that they help the persona seem like a real person. More general demographics, however, should always go on the back (they are not part of the persona proper – see Figure 3).

Figure 3, Suggested content for a minimal persona (front) and recruiting brief (back)

> **Collaborative:** Agile is collaborative at heart. Agile teams make hundreds of detailed design and planning decisions every day. If the core team does not appreciate or understand the user experience aspects of the project, it is likely to fail. (The popular alternative, of specifying the entire user experience in advance, before any code is written or any learning

has taken place is actually a waterfall approach, with all of the inherent dangers and drawbacks that waterfall methodology had in the 1980's and 90's[27].) The core team should be involved in the user research that is required for personas – at least as observers – and must be actively involved in the development of personas. There is substantial evidence that involving people in the decision-making process is essential if they are going to feel any sense of ownership of the resulting personas [28, 29].

One of the primary goals of personas is to create empathy and motivation for the team. Personas do this by allowing us to connect emotionally with other individuals rather than abstract collections such as 'users' or 'SingleTicketPurchasers' [30, 31].

3.2 Persona Stories

Persona stories differ from user stories in several important respects. Persona stories are written

- …about personas, not roles. Where it is useful or important to refer to roles, we simply qualify the persona name. Say we decide to characterize a warehouse returns operative as "Jack". In many cases we probably would not need to describe his role, since it would be obvious from the story. So a persona story would be as simple as:

  Jack processes a return

  Note that as a persona, it's Jack's behaviours and needs that are important. It may be that there are other Jacks in our system, probably working in a cold and dusty warehouse and without particularly good typing skills (these points would be part of the Jack persona). So Jacks might also process pick lists or label packages for dispatch.

1) …in the third person, that is, about the persona. So the form is

   <persona[:role]> <performs a task>[so that<unobvious goal>]

   The persona:role element is based on the Unified Modeling Language notation name:class. An alternative would be simply to place the role in parenthesis when it is needed. The optional 'so that' clause should only be provided if the goal of the task is not obvious. 'Julie adds an item to the shopping basket' does not really need an explanation.
2) …by user experience specialists in collaboration with business analysts and/or members of the core team. They are subsequently elaborated into scenarios and visual designs (again in collaboration with the core team) one or two project cycles ahead of their implementation [32]. This avoids the Agile/Waterfall conflict that is brought about by up-front UX design.
3) …after user research and rough design. The research is required to discover the needs and behaviours of users of interest to this project. Rough design defines the scope and shape of our venture. For example, if we are selling houses, we may decide that a comparative shortlist feature is more appropriate than a shopping basket and would write our persona stories accordingly.

One of the main benefits of persona stories, when produced as outlined above, is that they and their resulting scenarios and visual designs will be *descriptive* rather than *prescriptive*. That is to say that they describe interactions we have good reason to believe will work (because we have done research and evaluation with real users) rather than just prescribing what users must do. It is a subtle but extremely important difference (see Figure 4).

Figure 4, User stories are often prescriptive; personas stories should be descriptive (from research and evaluation)

About the Author

William Hudson has been building interactive systems for over 40 years, focussing on user-centered design for the past 20. He has written over 30 publications and is the creator of the Guerrilla UCD series of webinars (www.guerrillaucd.com).

William is the founder and principal of Syntagm Ltd, a small Oxfordshire consultancy specializing in user-centred design and training.

(For a more detailed biography see http://syntagm.co.uk/design/whudson.htm)

References

1. Wikipedia. *Chrysler Comprehensive Compensation System*. 2013 11/04/2013]; Available from: http://en.wikipedia.org/w/index.php?title=Chrysler_Comprehensive_Compensation_System&oldid=544135789.
2. Beck, K., *Extreme Programming Explained: Embrace Change*. 1999, Addison-Wesley.
3. Carroll, J.M., *Scenario-based design: envisioning work and technology in system development*. 1995, John Wiley & Sons.
4. Mills, C.W., *The Sociological Imagination*. 1959, Oxford University Press, USA.
5. Jacobson, I. *Object Oriented Development in an Industrial Environment*. 1987. New York, NY: ACM.
6. Wikipedia. *User Story*. 2013; Available from: http://en.wikipedia.org/w/index.php?title=User_story&oldid=548835190.
7. Cohn, M., *User Stories Applied: For Agile Software Development*. 2004, Addison Wesley.
8. Nygaard, K. *Program development as a social activity*. in *IFIP 10th World Computer Congress*. 1986. Dublin, Ireland: Elsevier Science Publishers.
9. Handy, C., *Understanding Organizations*, 2007, Penguin.
10. Ilgen, D.R. and J.R. Hollenbeck, *The structure of work: Job design and roles.* Handbook of industrial and organizational psychology, 1991. **2**: p. 165-207.
11. Constantine, L.L. and L.A.D. Lockwood, *Software for Use: A Practical Guide to the Models and Methods of Usage-Centered Design*. 1999, ACM Press.
12. ISO, *Human-centred design for interactive systems*, 2010, International Organization for Standardization: Geneva, Switzerland.
13. Wikipedia. *Scrum (development)*. 2013 12/04/2013]; Available from: http://en.wikipedia.org/w/index.php?title=Scrum_(development)&oldid=549332179.
14. Beck, K. and C. Andres, *Extreme Programming Explained: Embrace Change*. 2004, Addison-Wesley Professional.
15. Hudson, W., *Adopting User-Centered Design within an Agile Process: A Conversation.* Cutter IT Journal, 2003. **16**(10): p. 5-12. Available from: http://www.syntagm.co.uk/design/articles.htm.
16. Hudson, W. *Reduced empathizing skills increase challenges for user-centered design*. in *CHI 2009 and BCS HCI 2009 Conferences*. 2009. Boston, MA and Cambridge, England: ACM. Available from: http://www.syntagm.co.uk/design/articles.htm.
17. Ensmenger, N., *The computer boys take over: computers, programmers, and the politics of technical expertise*. 2010, The MIT Press.
18. Takeuchi, H. and I. Nonaka, *The new new product development game.* Harvard business review, 1986. **64**(1): p. 137-146.
19. Beck, K.e.a. *The Agile Manifesto*. 2001 12/04/2013]; Available from: http://agilemanifesto.org/.
20. Wikipedia. *Big Design Up Front*. 2013 12/04/2013]; Available from: http://en.wikipedia.org/wiki/Big_Design_Up_Front.
21. Strunk Jr, W. and E.B. White, *The elements of style*. 2008, Penguin (Non-Classics).
22. Nielsen, J. *First 2 Words: A Signal for the Scanning Eye*. 2009 12/04/2013]; Available from: http://www.nngroup.com/articles/first-2-words-a-signal-for-scanning/.
23. Google. *Search Results for "you are not the user"*. 2013 12/04/2013]; Available from: https://www.google.co.uk/search?q=“you+are+not+the+user”.
24. Polman, E. and K.J. Emich, *Decisions for Others Are More Creative Than Decisions for the Self.* Personality and Social Psychology Bulletin, 2011. **37**(4): p. 492.
25. Cooper, A., *The Inmates are Running the Asylum*. 1999, SAMS.

26. Ratcliffe, L.M., Marc, *Agile Experience Design: A Digital Designer's Guide to Agile, Lean, and Continuous*. 2011, New Riders.
27. Khalifa, M. and J.M. Verner, *Drivers for Software Development Method Usage.* Engineering Management, IEEE Transactions on, 2000. **47**(3): p. 360-369.
28. Sharot, T., *The Optimism Bias*. 2012, Robinson.
29. Jackson, S.E., *Participation in Decision Making as a Strategy for Reducing Job-Related Strain.* Journal of Applied Psychology, 1983. **68**(1): p. 3-19. Available from: http://libezproxy.open.ac.uk/login?url=http://search.ebscohost.com/login.aspx?direct=true&db=pdh&AN=1983-13804-001&site=ehost-live&scope=site.
30. Nordgren, L.F. and M.H.M. McDonnell, *The Scope-Severity Paradox.* Social Psychological and Personality Science, 2011. **2**(1): p. 97-102.
31. Sears, D., *The Person-Positivity Bias.* Journal of Personality and Social Psychology, 1983. **44**(2): p. 233–50.
32. Hudson, W. *User Requirements in the 21st Century*. Agile Record, 2012. 12-16. Available from: http://www.syntagm.co.uk/design/articles/userreq21c.htm.