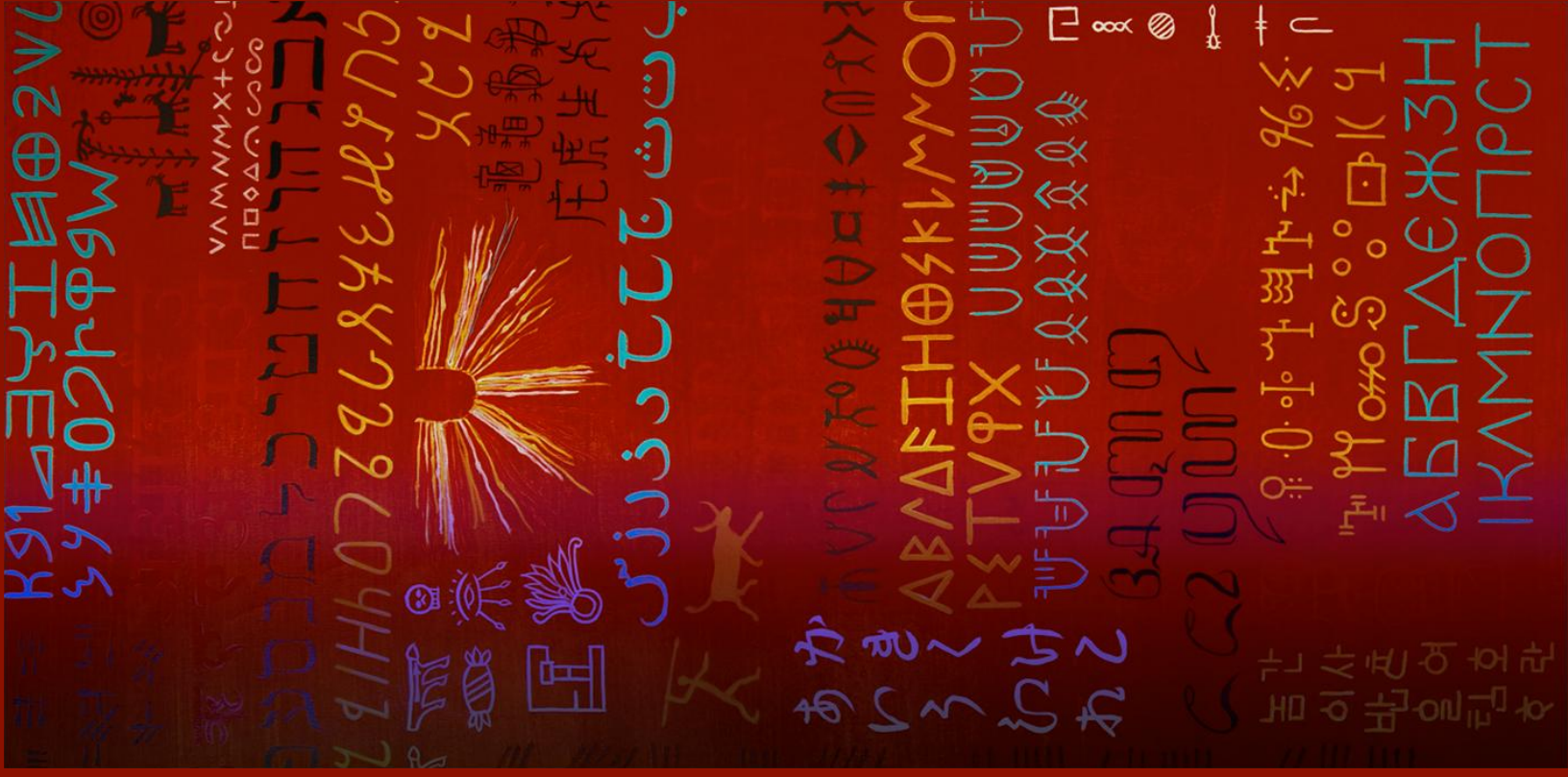

Agile Localization Practices

Recommendations

Tex Texin

Date: Oct 11, 2010

www.XenCraft.com

TexTexin@XenCraft.com

781 789 1898

Table of Contents

Introduction

There was an interesting panel at Localization World (Seattle, 2010): How Do We Create an Agile Localization Process That Can Keep Up with an Agile Development Process?

Panelists from Cisco and Welocalize presented their lessons learned and best practices.

It was a good session, but my takeaway was that their focus was on faster turnaround through automation, tighter communication, and throwing more resources at translation to keep up with the rapid development output. There are limits to the optimizations that can be performed to accelerate turn-around and these aren't enough to keep pace with rapidly iterating development.

The suggestions were generic optimizations that apply even in non-Agile situations. I asked about solutions specific to Agile. It was suggested that the Agile community should define and recommend Agile-specific solutions. I doubt this will happen. The localization industry should determine its own destiny.

It is ironic that the localization industry, an industry specialized in adaptation and customization, has been insistent on its existing process model and is not adapting itself.

Agile methodology should be embraced by LSPs and localization practices should be rethought in its context. This has the potential to lead to significant opportunities for innovative localization companies.

Scope

I should note that I am not an expert in Agile and every Agile-based project that I have had contact with, used a "hybrid" version. In other words, the team picked out the parts of the methodology they felt fit their needs and corporate culture. I don't know any one that uses "pure" Agile. These teams chose Agile specifically to improve project **transparency** and **predictability**.

For now, my comments are intended for Agile projects where most sprints are for internal or very limited client usage until beta or close to final release. There are some companies, such as the gaming industry, that have very short (a few days) release cycles, publishing immediately. There are similarities in that they both use rapid, iterative development practices, but going immediately to publication requires different tactics, which I may address in a future article.

What is Agile Methodology?

Agile breaks development projects into a series of small development "sprints" designed to show tangible progress and allow evaluation and redesign that can improve the end result. Although extended specification writing is eliminated, there is considerable planning and project management. Brief daily stand-up meetings are held for information sharing, project tracking, and commitment (re-)verification. It is important to have buy-in from all team members.

Key elements, relevant to localization staff:

1. The entire team, product management, development, QA, documentation, etc. has a seat at the table that plans and operates product development. This is essential to transparency, realistic scheduling and the collaboration needed for innovation and dynamic adaptation.

2. The sprints are designed to show demonstrable progress. At the end of each sprint the output should be functioning and testable i.e. something that can be evaluated.

3. Agile project management makes use of burn down charts and other techniques to track the work that is left to do, rather than the traditional approach of referencing the total work achieved to date.

4. Plans can change and are regularly updated to react to user feedback and lessons learned, These experiences lead to new (market or feature) opportunities and improved understanding of development or other costs.

5. As developers gain experience they recognize patterns and abstractions that give way to improved design and architecture. When this occurs, Agile encourages "refactoring" of the code. The cost of making significant changes to the code mid-project is repaid by faster future development, fewer bugs, and capabilities for features that would otherwise be difficult or expensive.

Agile Localization

How can localization fit into this scenario?

Let's first acknowledge the technical and brute force methods that can shorten turn-around. These would apply to any project methodology. For example:

- Increased number of translation staff.

- Machine translation.

- Replacement of file transfers with real-time notification of string creation/modification and direct access to the source repository, so translations can be provided continuously.
- WYSIWYG preview or other abilities to review or test translations in context.
- Controlled authoring.
    - Note that in an innovative, aggressively adaptive environment, controlled authoring is going to be perceived as inhibiting and counter-productive. I will return to this point later.

The following recommendations are more specific to the Localization industry stemming off of Agile's key elements:

Invest and Engage in Understanding Agile Methodology

This is an area of opportunity for the localization industry. With greater knowledge of Agile methodology, LSPs, tools vendors, localizers can participate in constructive ways, adding value and influencing changes that make sense for vendors as well as product teams. Of course, if industry participation is limited to echoing frustration with being able to keep pace with rapidly iterating development, as is currently the case, then the opportunity to influence will be lost.

Participate as Equals

Localization managers should have a seat at the project planning table for obvious reasons.

Localization directors (both client-side and vendor-side) should define new localization roles and responsibilities around Agile Methodology. A specialized Localization Project Manager (LPM) that is trained in Agile can be an important liaison and planning role. The LPM would provide collaboration, coordinate and innovate localization activities, and bring an appropriate localization focus to the Agile project and an Agile focus to the localization project. This would be a tremendous asset. With additional experience, other Agile-specific roles will become clear.

Localization vendors may not be adequately represented at the planning table by just the internal localization manager. When multiple LSPs are on a project, it is unlikely they can all participate directly in the planning without introducing chaos or adding considerable expense and overhead. Policies, practices, tools and incentives to drive vendor collaboration as well as acceptance of a common representative should be created. Coaching and guidelines for representatives, as well as vendors, on how to establish trust and encourage working together under an Agile umbrella should be developed (by the industry, the vendors, or the client…)

Sprint Planning

Sprints are designed to show demonstrable progress. They should produce functional, testable results. However, sprint output is not always a complete product. Earlier sprints will have missing features or use interim solutions so that there can be testable output until more comprehensive solutions can be made. Each successive sprint proves additional capabilities are working. Feedback from each sprint leads to changes to future sprint plans.

Localization should be incorporated into the product plan in the same way. The requirement is not to deliver every string of each sprint, translated into every language. The requirement is to show the progress of localization and to support the evaluation of the localized versions of the sprint output.

Localization management sitting at the Agile team table can drive collaboration on how localization can be segregated into sprints, what each sprint should prove and how overall project goals can be met, without attempting to be a complete localization of interim solutions. And without deferring localization until the end game…

Early sprints can prove software internationalization is adequate, that the process for creating kits and integrating and building localized kits is correct, and that translators have sufficient context and information to perform quality translations in the later sprints.

For example, in the early sprints machine translation **without** post-editing might be used. The translations might not be optimal, but the process would be proven and post-editing can then continue in parallel. Fully localized, manually edited language versions do not need to be delivered until later sprints.

Early sprints can also be used for glossary accretion, collecting key terms and initiating definitions and translations and refining them for consistency etc. and recommending improvements to the source text and the overall pattern and style of authoring.

As I noted above, it is difficult to gain acceptance for controlled authoring in a fast-moving innovative development environment. However, the planning team can agree that a particular sprint would be dedicated to source terminology revision based on linguistic assessment of prior sprints and the recommendations from the documentation and localization teams. The recommendations would improve consistency and reuse and perhaps also be a time for educating developers and authors on why certain terms were chosen and to evangelize keeping to the theme going forward (or perhaps to discuss why the terms and theme need modifications). Other milestones can be targeted at particular sprints (with acknowledgement that schedule revisions can occur.)

Project Management Tools

Management tools for localization that work analogously to burn down charts should be introduced. (E.g. Tracking number of untranslated strings remaining, rate for new string creation, etc.) These will improve transparency of the work remaining and highlight costs of revision, inconsistent authoring, etc. and identify areas where the localization process can be improved.

Planning For Rapid Revision

Agile projects have frequent revisions. The industry needs to rethink its tools and file formats, to support and frequent changes. Translation memory, localization kits, etc. should be designed to allow fast identification and removal of obsoleted strings, and have other support for quick updates and validation. Localization personnel need practices that support efficient and accurate communication about design and other changes. Where possible tools should catch errors by workers who forgot or missed the revision discussion.

Localization Refactoring

Developers refactor code to gain efficiency. Localization teams should also identify opportunities to gain efficiencies as patterns are discovered and lessons learned in the course of the project. For example, recommending use of particular sentence patterns, reorganization of strings, help, or localization kit contents, etc. to improve reuse or gain other efficiencies.

Localization managers are used to static, repeatable processes and reused roles going from project to project. Agile Localization Managers need to be more fluid and adaptable and to have workflow tools that support process changes, to accommodate the needs of the way development teams organize and schedule using Agile.

Conclusion

To improve support for Agile-based projects, localization needs to be part of the overall management decision making. Localization participants need to be trained in Agile so they can contribute knowledgeably and to use appropriate justifications to support the tasks demanded by localization efficiency. In addition, the localization process and tools need to accept rapid iteration and support removal of obsolete contents, rapid updating of refactored code, and more efficient management of localization data.

References

- agilemethodology.org
- Localization World Panel on Agile and Localization
- Additional details on software internationalization can be found at our I18nGuy™ and XenCraft™ Web sites.

Author Bio

Tex Texin is an industry thought leader specializing in business and software globalization services. His expertise includes global product strategy, Unicode and internationalization architecture, and cost-effective implementation and testing. Over the past two decades, Tex has created numerous global products, led internationalization development teams, and guided companies in taking business to new regional markets.

Tex is a contributor to several internationalization standards for software and on the Web. He is on the steering committees of IBM ICU and Globalsight open source products and the program committees for Unicode and other conferences.

Tex is a popular speaker at conferences around the world and provides on-site training on Unicode, internationalization, and globalization QA worldwide.

Tex maintains two Web sites for internationalization, the popular, instructional www.I18nGuy.com site and the site for his business www.XenCraft.com.

Tex is founder and Chief Globalization Architect for XenCraft. XenCraft provides global business consulting and software design, implementation, test and training services on globalization product strategy and software internationalization architecture.

Tex Texin
www.XenCraft.com
+1 781 789 1898
TexTexin@Xencraft.com