

# STeP-IN SUMMIT 2007

## International Conference On Software Testing

### The Distributed Agile Model

Written By

John Scarborough

VP, Quality Engineering

Presented By

Sonali Gogate

Aztecsoft-itest

Copyright: STeP-IN Forum and Quality Solutions for Information Technology Pvt. Ltd.

Published with permission for restricted use in STeP-IN SUMMIT 2007 in agreement with full copyrights from owner(s) / author(s) of material. All rights reserved. No part of this publication may be reproduced, stored in a retrieval system or transmitted in any form or by any means, electronic, mechanical, photocopying, recording or otherwise without the prior consent of the owner(s) / author(s). This edition is manufactured in India and is authorized for distribution only during STeP-IN SUMMIT 2007 as per the applicable conditions.

Produced By

**STeP-IN**  
Forum

[www.stepinforum.org](http://www.stepinforum.org)

Hosted By



[www.qsitglobal.com](http://www.qsitglobal.com)

---

## *The Distributed Agile Model*

Since proponents of Agile processes<sup>1</sup> prefer face-to-face conversation as the means for communication within a development team, “distributed Agile” - in which members of an Agile team are situated in more than one workplace -- may seem unlikely.

But Agile processes are not rule-bound; rather, they optimize for methods and dynamics by which organizations are able to build, maintain, and apply systems of knowledge within changeable, context-dependent, collaborative work situations.

So when we evaluate the viability of distributed Agile (DA), our concern is not whether it complies with a particular definition of Agile, but whether it can support iterative, incremental, and sustainable development; promote teamwork; promote self-management; adapt readily to emergent changed requirements; and increase customer participation in the development process. Based on my experience I would say DA is, in this sense, viable. But special measures must be taken, especially in the area of communication. I will briefly describe those measures in this paper.

First, though, it may be helpful to state some background. We worked with groups who, following their companies' recommendation or mandate, had adopted Agile project management, but also wanted to outsource their testing. We worked with start-ups who were committed to Agile models for development and wanted to outsource within that model because they had seen such extraordinary improvements in productivity and code stability. We also worked with companies who were trying out whatever they thought might work better than the cause of their current headache.

I wish I could say that, for developing a website using Ajax, for example, you do or you do not want to attempt DA. Or that if you are working on the first release of a

---

new product, you should stick to single-sited Agile. In fact there is very little correlation between the type of project and the success or failure of DA. Failure is almost certain however if team members do not “get it”, that is do not understand that Agile is not about freedom from documentation, or freedom from process (although it very well may be free from these), but is about producing working software at short intervals using iterative processes that adapt to emergent conditions without losing their purpose.

Some varieties of testing are a very good fit for DA, such as running full regression suites on the most recent known-good build, performance testing, automation development, and end-to-end testing. The home group (that is, the group that owns contact with company management) works side-by-side with the developers, perhaps reviewing acceptance tests and designing testcases that expand upon tests developed in test-driven development, and the remote group covers non-functional testing.

One of the best fits for DA has been a sort of cluster of remote scrums, where the home group drives development of the core engine, while a remote group is spawned whenever an adapter or significant new feature is required, each remote group communicating closely with one member of the home group who attends daily standups etc. I don't recommend trying this without doing due diligence; but it is useful to know that DA does work in a variety of situations.

Here are a number of practices, I wouldn't want to call them “best practices” because I am certain they can be improved upon. Their strongest recommendation is that they have consistently worked in a variety of projects deployed in a DA model.

---

## MAKING DISTRIBUTED AGILE WORK

- *Keep an inclusive test plan*

Even in straightforward Scrum, a backlog or schedule or even a set of cards on the wall reminds everyone what tasks the team owns, i.e. must accomplish in the current sprint (or series of sprints). This is of even greater importance for DA; if there are multiple remote groups, while the “scrum of scrums” structure is useful, it is not sufficient - the remote group will need something to refer to in case the home group is unavailable.

- *Keep the customer engaged with both remote and home groups throughout the project.*

The temptation is to maintain contact with the remote group only, but that would damage the remote group’s sense of belonging to a team, which is the basis for communication. The customer should at least participate in conference calls.

- *Coordinate by “scrum of scrums”*

Represent each remote group in home group through participation by one home group member who is in daily telephone contact (if personal contact is impossible) with the remote group. Participation of remote groups by conference call alone provides much weaker linkage.

- *Exchange remote and home group members within the team*

Ideally, remote group members visit the home group for several days, and home group members visit remote groups. Even if only one person from the home group visits a remote group for a week or two, all members of both groups will tend to bond. This is especially useful if the two groups are from markedly disparate cultures..

---

- *Communication protocols*

Since decisions regarding adoption of communication and reporting protocols have a much greater impact in DA than in single-site Agile, they should be discussed as part of pre-sprint planning, and any required HW or SW resources obtained well in advance of sprint kick-off.

- *Defect management*

For large-scale projects using multiple Agile teams, while each team tracks defects and their resolution during the sprint using a backlog or programmatic equivalent (e.g. Rally), at the end of the sprint all fixed, resolved and unresolved bugs should be moved into a defect database (e.g. Bugzilla) that tracks all bugs across the project/product so that there are no loose ends resulting, for example, from the remote group handling regressions; and so that trending can be identified across multiple groups and teams. While it is tempting for the remote group to enter all bugs into the master defect database during the sprint, that defeats the task-based focus of Agile.

- *Staff assessment*

Since every project has different resource needs, ensure that team members have the needed skill-sets and experience. The team that put together version 1.0 may not have what's needed for the features planned for 1.1. Somehow remote groups are especially prone to issues of "inheritance", where resource qualifications are not re-assessed on a sprint-by-sprint basis.

- *Trust and feedback*

Because DA reduces communication bandwidth, misunderstandings are rampant. Trust is critical to success. Trust enables the exchange of constructive feedback and criticism. Establishing common non-work interests

---

helps a lot. The best social glue I've seen was when members of both the home and remote groups followed World Soccer championships.

- *Access to resources*

Give everyone equal access to the same resources. Wikis are helpful because their topic hierarchy can be changed on the fly, links can be shared instantly, etc.

- *Avoiding process inheritance*

Because more process is required by DA to keep home and remote groups in synch, processes are likely to be carried over from one project to another. They will lose their "appropriate process for the project" attributes, adding unnecessary baggage. While keeping everyone on the same page, it doesn't hurt to maintain a healthy skepticism for all process, verifying its need before adopting it.

- *Documenting test results*

Although working code is an indicator of some degree of accomplishment, it is never a replacement for documentation necessary to record what tests have been completed, such as non-functional testing by the remote group in areas like security and compliance.

- *Looping business analysts in to specify documentation*

Agile has a tendency to reduce documentation to zero. Don't assume that the Agile team manager or the team itself are capable of deciding how much documentation is required -- bring business stakeholders into the pre-sprint discussion.

If a team realizes benefits in cost and quality while using Agile, it shouldn't assume that every aspect of Agile is responsible for its success. The key may be

---

iteration alone, as seemed to be the case in NASA's Project Mercury.<sup>2</sup> Success may have been due merely to keeping management out of the picture while coding.

DA requires a little more coordination, regulation, and scheduling -- in other words a little more discipline. Unless an Agile team is composed entirely of bright, experienced, self-managing, team-supporting multiple-domain experts, you'll need some degree of discipline even if everyone sits around the same table for the entire sprint. DA is the same; but moreso.

---

<sup>1</sup> <http://www.agilemanifesto.org/principles.html> (viewed December 1st 2006).

<sup>2</sup> Iterative and incremental development (IID) was known and practiced at least as early as the 1960s in NASA's Project Mercury, based on ideas of a quality expert named Walter Shewhart at Bell Labs in the 1930s. See <http://www2.umassd.edu/SWPI/xp/articles/r6047.pdf> (viewed November 15th 2006)