# Fully Distributed Scrum

## Linear Scalability of Production between San Francisco and India

Jeff Sutherland, Ph.D.
Scrum Training Institute
Boston, USA
jeff@scruminc.com

Guido Schoonheim
Xebia B.V.
Hilversum, The Netherlands
gschoonheim@xebia.com

N. Kumar, V. Pandey, S. Vishal
Xebia IT Architects India P.L.
Gurgaon, India
nkumar@xebia.com

***Abstract*—The Scrum software development framework was designed for the hyperproductive state where productivity increases by 5-10 times over waterfall teams and many co-located teams have achieved this effect. In 2006, Xebia (The Netherlands) started localized projects with half Dutch and half Indian team members. After establishing a localized velocity of five times their waterfall competitors on the same project, they moved the Indian members of the team to India and showed stable velocity with fully distributed teams. The ability to achieve hyperproductivity with distributed, outsourced teams was shown to be a repeatable process and a fully distributed model is now the recommended standard when organizations have disciplined Scrum teams with full implementation of XP engineering practices inside the Scrum.**

**Previous studies used overlapping time zones to ease communication and create a single distributed team. The goal of this report is to go one step further and show the same results with team members separated by the 12.5 hour time difference between India and San Francisco. If Scrum works without overlapping time zones then applying it to the mainstream offshoring practice in North America will be possible. In 2008, Xebia India started engagements with partners like TBD.com, a social networking site in San Francisco. TBD has an existing core team of developers doing Scrum with an established local velocity. Adding Xebia India developers to the San Francisco team with a Fully Distributed Scrum model achieved linear scalability with a globally distributed outsourced team.**

***Keywords-scrum; distributed; outsource***

## I. INTRODUCTION

Co-located teams are more productive than distributed teams, often doubling productivity over teams distributed within the same building [1]. Loss of productivity increases with globally distributed teams, with outsourced distributed teams, and when scaling the size of teams.

"If there are n workers on a project, there are $(n^2-n)/2$ interfaces across which there may be communication, and there are potentially almost $2^n$ teams within which coordination must occur. The purpose of organization is to reduce the amount of communication and coordination necessary; hence organization is a radical attack on the communication problem." Fred Brooks [2]

Brooks asserts that adding more people to a late project just makes it even later because of these communication effects and because productivity per developer always decreases on waterfall projects when team size increases. Historical attempts of a "radical attack" on the communication problem have universally failed.

Coplien proved in hundreds of case studies during the Bell Labs Pasteur Project that "communication saturation" is directly correlated with high productivity in software development. Face to face communication in a cross-functional team can increase productivity 50 times over waterfall teams [3]. His studies led the first Scrum team to implement daily meetings and performance over 20 times average waterfall performance was achieved with some teams [4]. The "radical attack" of Scrum on the communication problem works for co-located teams, but can it work for distributed teams?

This paper documents a model for producing distributed and offshored team velocity that is equal to co-located velocity of a single team with a 12.5 hour time zone difference. The model is repeatable and is recommended for teams that can execute a high performance Scrum implementation with XP engineering practices inside.

## II. CHALLENGES IN OFFSHORE OUTSOURCING

U.S., European, or Japanese companies often outsource software development to Eastern Europe, Russia, or the Far East. The three key advantages that offshoring strives to achieve are (1) lower costs of labor, (2) capture talent not available locally, and (3) increase and decrease project size without layoffs.

Offshore costs are typically about 30% of onshore costs per developer hours (ignoring communication overhead which doubles costs on average). Xebia onshore teams run at 5 times the velocity of a waterfall team. Offshoring to a waterfall team could raise the cost of a project by 150% due to low velocity of waterfall teams.

Capturing talent is difficult in India and China as turnover rates on projects are 30-50% a year. Xebia has shown that an agile development environment combined with good management can reduce this turnover rate to less than 5%.

Decreasing team size offshore with typical distributed team models results in loss of critical knowledge. Fear of this effect often causes vendor lock in. The fully distributed model retains all knowledge onshore as team knowledge resides in all locations. Thus the benefits claimed for offshoring are actually achievable.

## III. Regular Distributed Team Models

Most offshore development efforts use a degenerative form of the Isolated Scrums model where outsourced teams are not cross-functional and not agile. Requirements may be created in the U.S. and developed in Dubai, or development may occur in Germany and quality assurance in India. Typically, cross-cultural communication problems are compounded by differences in work style in the primary organization vs. the offshore group. In the worst case, teams outsourced this way are not using Scrum and their productivity is typical of waterfall projects further delayed by cross-continent communications lag time. Implementations of Scrum in a data rich CMMI Level 5 company simultaneously running waterfall, incremental, and iterative projects, showed productivity of Scrum teams of at least double that of waterfall teams, even with CMMI Level 5 reporting overhead [5]. Outsourced teams not using Scrum will in the best case achieve less than half the velocity of an onshore site using Scrum assuming equal talent across teams.

Best practice recommended by the Scrum Alliance is a Distributed Scrum of Scrums model. This model partitions work across cross-functional, isolated Scrum teams while eliminating most dependencies between teams. Scrum teams are linked by a Scrum-of-Scrums where ScrumMasters (team leaders/project managers) meet regularly across locations. This encourages communication, cooperation, and cross-fertilization and may be appropriate for newcomers to agile development or those who have offshore limitations that cripple the productivity of the fully distributed model.

## IV. Fully Distributed Scrum Model

Fully distributed Scrum teams are cross-functional with members distributed across geographies. This means that a single team will have members in multiple locations. A single team might have four developers onshore and four developers offshore. These team members share a single sprint backlog and share code ownership. In the SirsiDynix case, the Scrum of Scrums was localized with all ScrumMasters in Utah. At Xebia, ScrumMasters may be in the onshore or offshore depending on project needs.

Xebia's Fully Distributed Scrum model has all teams fully distributed and each team has members in multiple locations. While this "OneTeam" model might seem to create communication and coordination burdens, most communication is handled by following the Scrum cycle. The daily Scrum meetings actually help to break down cultural barriers and disparities in work styles while simultaneously enhancing customer focus and offshore understanding of customer needs. On enterprise implementations, it can organize the project into a single whole with an integrated global code base. Proper implementation of OneTeam provides location transparency and performance characteristics similar to hyperproductive co-located teams.

## V. Xebia TBD.com Case Study

In previous papers we illustrated the use and performance of Fully Distributed Scrum [6, 7]. These case studies concerning Fully Distributed Scrum utilize an overlapping time zone. Shared time during the workday helps in creating a OneTeam situation and eases communication. However, large parts of the offshoring industry do not have this overlap available. Notably, the United States and India do not have this overlap.

In order to demonstrate the positive effect of Fully Distributed Scrum on large time difference situations we will show a project that Xebia India worked on with a company located in San Francisco.

This company, TBD, maintains a social networking website, http://www.tbd.com/. Whereas sites like MySpace and Facebook target a young audience, TBD differentiates itself by targeting an audience of > 40 years. TBD stands for 'To Be Determined', illustrating that life has a lot to offer to those of higher maturity. On TBD.com people meet online to discuss topics such as raising teenage children, life philosophy and work/life balance.

The company standardized development on Scrum in an early stage and created a live website with an active community. After this initial product creation TBD wanted to scale up development for a period of time with an offshore partner that could match their Scrum process.

Xebia and TBD started with a six week pilot and continued working together on a successful project for a period of eight months. After this period the required boost in functionality was achieved and the economic recession started. The project was completed to the satisfaction of TBD and development was reduced again to a small local team.

### *A. OneTeam engagement model*

Central to Xebia's approach to offshoring is the notion of distributed teams (OneTeam). A distributed team has team members on multiple locations where all team members are committed to the same sprint backlog. TBD and Xebia used this approach to set up a single distributed team consisting of TBD engineers in San Francisco and Xebia engineers in New Delhi.

This setup mixes engineers from two different companies coming from different cultures separated by a huge time difference into a single team. While this seems daunting at first, this OneTeam model actually proves to be essential in bridging the gap between the United States and India.

### *B. Quick project setup*

A short period of co-location is very useful and recommended in order to be effective and up to speed as soon as possible with a distributed project. TBD's Product Owner and Scrum Master/Technical Lead traveled to India and spent two weeks at Xebia India. During this period, the main objectives were to get to know each other, to setup a functional work environment, to agree on ways of working, to transfer knowledge, to share short and medium term business goals for TBD, and to setup success criteria and measurements for the collaboration.

Prior to this visit documentation and codebase access was provided to the Indian team members to have an overview of the project and thus be able to make best use of

the co-location period. This preparation proved very useful in achieving the quickest possible startup.

Initially, Xebia India started with three developers, matching the existing team in San Francisco. The team shared a common code base repository on both sides, used the same wiki and Bug Tracking tools, agreed on a common definition of DONE (see Agile Practices and Tools) and shared and worked from one Sprint Backlog. The personal relationships formed during this visit are important to facilitate easy communication with the Product Owner and onshore team members.

In order to come to clear working agreements the team held a 'Norming and Chartering' session. This is a Xebia best practice where very specific agreements are made in the fields of practices, (coding) conventions, tooling, process, evaluation & escalation, and anything else that the team would like to establish.

During the TBD visit, the team immediately started the first shared iteration. Iteration length in this project was three weeks. Focus was put on knowledge transfer as the existing codebase was substantial and functionality often involved modifications. In order to get the best velocity as well as maximum knowledge transfer, the Product Owner selected stories that would facilitate knowledge build-up. The Indian team members used the presence of TBD staff in India during the first two weeks of the sprint to complete these stories and were able to continue with enough knowledge afterwards. This was also due to their study of the system and source code prior to the start of the project. In addition, the Indian team members put in extra work hours during the initial two weeks to get the most out of the onsite presence of TBD. Due to these measures the team velocity doubled immediately when the number of developers doubled, despite the need for knowledge transfer.

To evaluate the engagement between TBD and Xebia a number of points were evaluated every iteration. Velocity was measured against expected velocity. The capability of Indian team members to work independently was subjectively evaluated. New modules and areas that the Indian team members worked on were judged on quality. The entire team measured general team dynamics on a subjective scale and completed a quality questionnaire. Combined with the more free format retrospective, this gave very good insight into current progress and into subjective feelings about the project. This facilitated very open and direct feedback, leading to better team building.

Scrum Master, Product Owner and testers in this project were localized in the US with developers spread out over both locations.

### C. *Scrum cycle pulls the team together*

The most important ingredients for building a team are communication, shared vision, active participation, shared ownership and shared goals. The Scrum process is excellent for providing all of the above if you are in the same team working off the same sprint backlog. Certain modifications to the Scrum cycle are necessary in order to achieve the same feeling of OneTeam when the time difference is so large (12.5h).

All larger Scrum meetings are shared by the whole team and done jointly using videoconferencing. This requires team members on both sides to be flexible in creating overlapping timeframes for these meetings. In order to minimize the time required, the Sprint planning is prepared separately with both sides of the team by the Product Owner. Sprint planning 2, creating tasks, is performed jointly after both sides have received explanation of the user stories. Daily standup meetings are not shared by all team members as this would put the whole team into permanent fatigue and seriously impact performance. Instead the rotating role of proxy was created: someone working odd hours and attending both standup meetings. The choice was made to base the proxy in the US and not in India because of available knowledge. The US proxy could answer many questions directly. The US proxy shifted part of his work day to the evening (approx. 8PM) and the India team started a bit early (approx. 8AM) to create the overlap required for the standup meetings. Different individuals would take the proxy role on different days, allowing different people from both sides to regularly interact directly. Once a week, a fully shared standup was held in addition to this rhythm to allow all team members to see each other 'face to face' and exchange first hand information. It was also used for design discussions between developers from both locations, to get feedback from testers and to share any new stakeholder information. This meeting proved essential to bridge any gaps in terms of information sharing across locations. As with earlier projects regular videoconferencing aided team building substantially.

The demo and retrospective are done together with videoconferencing. At the end of the day team members on both sides wrote an update on the wiki about what they accomplished that day and about any current impediments. This approach was chosen over daily digest mails as it offers one single location for all updates along with easy history.

This modified Scrum cycle covers virtually all communication. It is both the minimally required ceremony and the most effective way to connect the distributed team members to form a team. That makes it the best approach to offshoring currently available.

### D. *Linear productivity*

As sections III and IV explain, traditional outsourcing using a waterfall approach has proven to be extremely costly. Fully Distributed Scrum eliminates the usual waste in communication overhead that traditional offshoring brings and establishes a strong sense of shared ownership and clear purpose with full transparency. In short it succeeds in bringing Fred Brooks definition of a "radical attack" [2] on the communication problem to distributed development.

The TBD project actually demonstrates a slight increase in productivity from the first sprint. The Indian team members are experienced and senior enough to be quickly up to speed and the technology stack used by TBD is familiar. Handling knowledge transfer was explained in the section about quick project startup.

Figure 1 shows the velocity in story points over time averaged per person. The Indian team members joined in iteration 41.

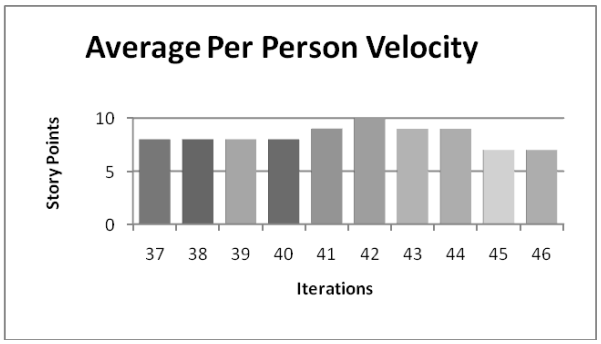


Figure 1. Average per person velocity

Iteration 41 has almost double the development capacity of iteration 40. In iteration 41 a total of 45 story points was accomplished versus 24 story points in iteration 40. During this transition the performance per person remains roughly the same with even a slight increase.

This constant performance afterwards shows linear scalability, instead of the traditional dip in performance in offshoring. This linear scalability in performance is similar to results delivered by good co-located teams.

### *E. Delivering high quality*

Measuring of the amount of issues found shows consistent quality in development after scaling up. At the beginning of the project an audit of the existing codebase was done. Areas prone to break during refactoring were identified and addressed. The Indian team members had previously worked on large hyperproductive enterprise projects and had a number of valuable suggestions to improve upon technical quality and definition of done.

One major step taken after initial setup is better integration of testing work into the sprint. Using previous experiences of the Indian team members, the time to go to production after release was reduced from a week to a single day. This greatly improved the flow of development.

Starting from the second sprint, the Indian team members managed to pick up substantial functionality on their own, after three months they were fully knowledgeable, and during the summer holidays they took over development and support all together. Refactoring initiated by the Indian team members included introducing new patterns into the architecture.

### *F. Agile practices and tools used*

The TBD project adopted the full set of XP practices. During the engagement, the team increased the rigorousness with which XP practices were followed, inspired by the previous experience of the Indian team members.

The shared definition of done included items such as proper measured test coverage, passing of all previous existing tests, verifying continuous integration, testing by the testers on the team, and updating project status information.

The San Francisco team members had to adjust their way of working to allow smoother work in a distributed manner. Examples are more clearly defined user stories, a defined ready state for requirements to be allowed into the sprint, and clearer definition of done. These changes helped to ease development and testing and helped the Product Owner to more accurately translate business goals into user stories.

Digital online tooling was used for collaboration. The product backlog and sprint backlog were managed with a digital Scrum tool (Pivotal Tracker). A separate issue tracker (JIRA) was used for managing bugs. The team relied heavily on a wiki (Confluence) for requirements specification and system documentation as well as team information like status updates. Videoconferencing was done using Skype with Adobe Acrobat Connect for desktop sharing.

### *G. Business case involvement*

Xebia had the opportunity to present the project to Lean & Agile Software Development authorities Tom and Mary Poppendieck [8] who visited the Xebia offices in India. They had a number of valuable suggestions in the area of business case involvement.

As a result, the team members in India broadened their focus to include the business impact of the features they delivered. The sprint demos were initially localized in the US. This was changed to provide the Indian team members with maximum direct feedback on their work. The Product Owner decided to share progress reports to TBD management with the whole team. This added strongly to the OneTeam feeling.

TBD shared usage information from Google Analytics and feedback from power user panels with the team members to give real time insight into the progression of their KPI's. The team members all used accounts on the live system to double check their features in production and to validate the user experience. Once they got more involved in the actual business, they started researching competitors and could suggest functional improvements. This resulted in a number of significant changes to the site, such as a revamp of the Friend Finder / Invitation functionality, which is essential to the growth of a social networking site.

Success for TBD is measured in an expanding user base and increased usage. Good product ownership is geared towards increasing this value. Figures 2 and 3 show that

during the time of the engagement, both number of users and page views per user visit quadrupled.

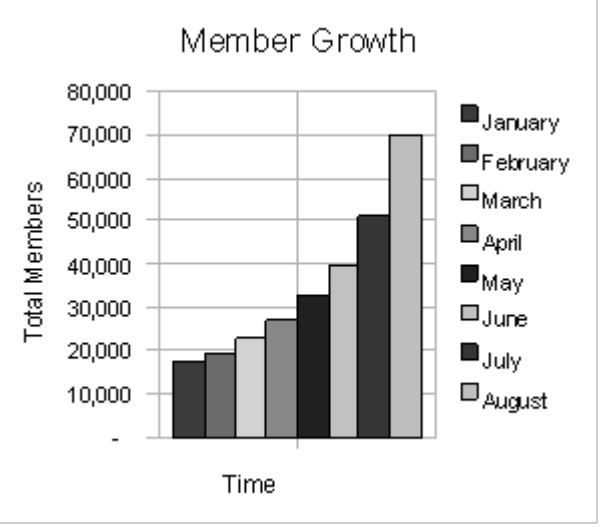


Figure 2. Member growth during TBD project

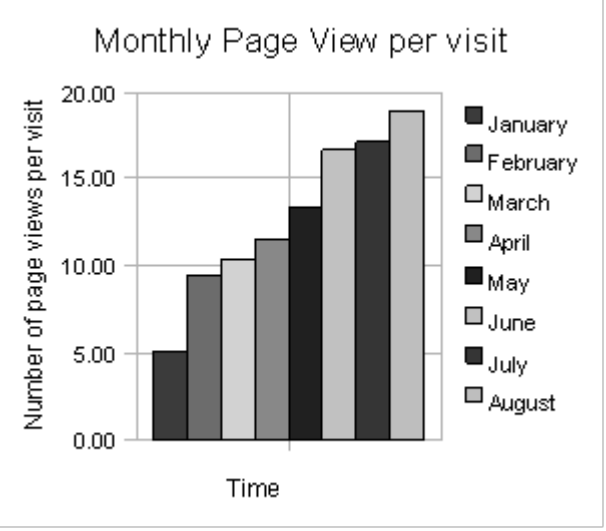


Figure 3. Usage growth per visit during project

### H. *Customer success factors*

The main driver for TBD to engage in offshoring was cost savings. As productivity was equal to local productivity with the same quality but at a lower cost this benefit was accomplished. The second benefit is the availability of talented and skilled knowledge workers. TBD had access to the full talent pool of Xebia for consulting and added services such as Agile consultancy and graphic design. Thirdly, due to the time difference it was possible to set up 24/7 support staffed by the development team without much overhead. During the US nighttime the support calls were routed to the Indian developers and vice versa.

Finally, flexibility with knowledge retention was a key benefit. At the end of the project, TBD was able to downscale without loosing any crucial knowledge of the system. Flexibility in sourcing is actually a recurring main driver for Xebia's clients to engage in offshoring.

This shows all key advantages of offshoring were achieved.

### I. *Project challenges*

A number of difficulties remained throughout the project and were not completely overcome. Firstly, distributed demo's were at times hard to organize since they involved many business stakeholders that were not always available in the early morning or later in the evening, creating logistical issues. Secondly, some activities like shared distributed design and modeling were limited as the team had very limited overlapping time. Thirdly, due to budget constraints no travel was possible after the initial setup. This meant that the offsite team members did not experience the environment and context of the onsite team members. Traveling during the project is a Xebia best practice: It strengthens personal relations and context awareness. And finally, some work simply had to be done locally. For instance, performance tuning involved cooperation of a US based hosting party and had to be done locally, although a specialist consultant from Xebia was able to add expertise. For testing the resources were located in the US. At times this was not ideal due to the time difference delaying feedback.

## VI. CONCLUSIONS

While previous publications and a wide range of Xebia projects have shown hyperproductivity in distributed teams, this is the first documented case dealing with maximum time difference. In summary, it is proved possible to create a distributed/outsourced Scrum with the same velocity and quality as a co-located team without overlapping time zones using talent from two different companies.

The OneTeam strategy along with the modification to the Scrum cycle described in this paper lowers costs, captures offshore talent, and has the ability to increase and decrease team size without loss of knowledge. TBD's ability to downscale at the end of the project demonstrates this ability.

Fully Distributed Scrum is the recommended strategy to unlock the full potential of Indian offshoring for the US market for teams capable of fully implementing the practices of Scrum and XP.

## VII. REFERENCES

[1] S. Teasley, L. Covi, M. S. Krishnan, and J. S. Olson, "How Does Radical Collocation Help a Team Succeed?," in *CSCW'00* Philadelphia, PA: ACM, 2000, pp. 339-346.

[2] F. P. Brooks, *The Mythical Man Month: Essays on Software Engineering*: Addison-Wesley, 1995.

[3] J. O. Coplien, "Borland Software Craftsmanship: A New Look at Process, Quality and Productivity," in *5th Annual Borland International Conference*, Orlando, FL, 1994.

[4] J. Sutherland, "Agile Can Scale: Inventing and Reinventing Scrum in Five Companies," *Cutter IT Journal,* vol. 14, pp. 5-11, 2001.

[5] J. Sutherland, C. Jacobson, and K. Johnson, "Scrum and CMMI Level 5: A Magic Potion for Code Warriors!," in *Agile 2007*, Washington, D.C., 2007.

[6] J. Sutherland, G. Schoonheim, and M. Rijk, "Fully Distributed Scrum: The Secret Sauce for Hyperproductive Offshored Development Teams," in *Agile 2008*, Toronto, 2008.

[7] J. Sutherland, A. Viktorov, and J. Blount, "Adaptive Engineering of Large Software Projects with Distributed/Outsourced Teams," in *International Conference on Complex Systems* Boston, MA, USA, 2006.

[8] M. Poppendieck and T. Poppendieck, *Lean Software Development: An Implementation Guide*: Addison-Wesley, 2006.