When the VP is a Scrum Master, You Hit the Ground Running

C. Lynn Cowan
Unisys Corporation
Irvine, California, USA

Abstract - Companies adopting Scrum as a software development framework often start with a pilot project and slowly grow adoption across the organization. Unlike this typical incremental implementation of Scrum, Unisys Cloud Engineering adopted Scrum throughout the entire organization at the same time. The adoption was initiated and supported by upper management. This paper describes how the effort started, provides an assessment of why this method worked, and details lessons learned that could benefit others considering this rollout method.

Keywords - Scrum; Scrum Master; implementation; adoption; management

I. INTRODUCTION

The first time I heard of Scrum I was in a presentation given by a project manager in another organization. I was at my desk listening and as soon as he said, “two-week development cycles,” I tuned out and went back to reading e-mail. Two weeks? More like two years! That was crazy talk. Fast forward a couple of years, a lot of layoffs, a wholesale management change, new product direction, and there I was sitting with the Cloud Engineering management team in a Scrum Master’s class.

In early 2010, Unisys Cloud Engineering was a newly crafted organization of senior engineers, architects, and managers with decades of Waterfall development on our resumes. One year after that Scrum Master’s training, we were amazed that we had completed over 200 sprints for 20 complex system software products across four geographies, including Bangalore. And did I mention that during that time one development site moved buildings and we landed a huge new account? While every organization’s Scrum adoption story is unique, ours offers insights into whole organization Scrum adoption, what it took to get it going, what we learned in the process, and why you might consider doing the same.

II. ON YOUR MARK

The consideration for adopting Scrum all at once for a new organization was a decision made by a few key leaders at Unisys. Due to rapid changes in cloud technology, senior leaders understood that we couldn’t do business as usual. Executive sponsorship was key to getting started.

Executive sponsorship. Unisys had been building elements of a cloud strategy for several years, but Bob Supnik, VP of Unisys Engineering and Supply Chain, wanted someone to create and drive the vision. So he hired Mark Hodapp as VP of Cloud Engineering. Not only did Mark have experience in cloud technologies, he was a Certified Scrum Master who had delivered products using the Agile Scrum framework. Supnik knew that Unisys Engineering needed to be agile to compete in the cloud marketplace. As a result, he agreed to sponsor Hodapp’s proposal to adopt Scrum and prototype its use for the Cloud Engineering organization. With sponsorship from executive management and encouragement from our marketing and sales organizations, Cloud Engineering was chartered to prototype and scale the effort to see if an Agile approach to development could be achieved and sustained. At least two other Unisys development teams had experimented with Scrum, but this was the first large, top down effort launched.

III. GET SET

While sponsorship is critical, it is not enough to ensure implementation goes smoothly. Focused groundwork must be in place to make it successful.

Creating a top down Scrum organization. The Cloud Engineering organization was crafted from existing product teams with expertise in hardware, software, networking, and security. It was an opportunity to design an organization based on the separation of powers advocated by Scrum principles. Hodapp grouped Product Owners in technology-based Grand Challenges; they report to a Lead Product Owner who sets priorities for the organization. A parallel structure of architects serves each Grand Challenge and product team. Architects are responsible for ensuring that product designs align with the overall cloud architecture that they define together with the Lead Architect. Engineers in the four development centers (Bangalore, California, Minnesota, and Pennsylvania) report to a co-located People Manager. Working with Product Owners, People Managers form and adjust teams, coordinate training, and manage equipment resources. Each location also has at least one Scrum Master responsible for evangelizing the Scrum process.

Training management. To speed adoption of Scrum, Hodapp opted for a wholesale rather than incremental approach. Despite severe travel budget restrictions, he got approval to assemble the whole management team at a development site for BigVisible, a consulting company, to conduct four days of Scrum Master and Product Owner
training. Although most of the team had worked together before, some for decades, many were meeting in person for the first time. This investment not only helped the new management team bond, but it gave them a common understanding of Scrum principles and the roles they were to play that would drive consistent implementation of Scrum throughout the organization.

Plan for distributed teams. Our Scrum training emphasized the benefits of co-located teams and artifacts such as story cards that the team moves from column to column on a story board as work proceeds through a sprint. With four development centers and a management directive to assign the people with the right skills where needed regardless of location, we knew we needed distributed Scrum teams to work effectively right from the start. Even the few teams where engineers were co-located would benefit since it seemed that someone was always working from home due to an early customer call or a sick child. It was also inevitable that every team would need skills available only in another location.

Fortunately, one team that had been using Scrum had already evaluated web-based Scrum tools because the Product Owner and team were in different cities. The Product Owner had written a simple, flexible tool to manage stories and tasks. The Monday after our class, the Cloud Engineering management team decided to adopt this internal tool because of its low cost, immediate availability, intuitive interface, and ability to customize it quickly for our specific requirements. It also satisfied our VP’s requirement for complete transparency so that all product backlogs, story details, burn down charts, and even tasks could be viewed by other teams, stakeholders, upper management–anyone in the global Unisys organization. Rolling out a common tool within a week of our class was critical to the speed with which we became effective distributed Scrum teams.

In addition to ensuring that Product Owners and teams knew how to use our Scrum tracking tool, Scrum Masters shared best practices on how to make distributed Scrum meetings more effective. Using teleconferencing and Live Meeting to display reports, Scrum Masters took roll at each meeting, ensured that during Scrum each team member spoke in turn and could be heard, and explained comments or described body language to ensure that remote team members felt included.

IV. GO

No one can tell you how to get started with Scrum. Applying Scrum principles, you time box your planning and training and just begin. You’ll know more from doing than you ever could from thinking about what you could or should do.

Just begin. After the training, we returned home and wondered “how do we start?” The management team who had been trained began functioning in our new roles. Product owners met together to define user types for our product suite and converted requirements into stories. People Managers evaluated and refined existing teams and created new teams for new initiatives. Architects began meeting together to define high level architectures, and also met with product teams to define and align product designs. Hodapp ran the weekly staff meeting in a daily standup format to reinforce the principles. Each staff member in turn states accomplishments of the prior week that others need to know about and his or her plans for the next week. Any impediments reported are tracked week to week until they are resolved. Using this format, even staff members who are not active members of Scrum teams can practice what they learned.

Train the engineers. The Scrum Masters were tasked with introducing the rest of the organization to the Scrum framework. We distilled the four-day class into a two-hour overview presentation, which a Scrum Master gave at each development site. The intent was to give engineers enough to get started with basic techniques. Over time, the Scrum Masters were expected to evolve engineers’ understanding of Scrum principles and rationale as everyone gained experience with the framework.

Existing teams. Teams who were in the midst of critical releases started midstream by breaking their current work into sprints and starting daily standups. Product Owners converted requirements into stories that the teams sized and broke down further into tasks.

New teams. New teams that were chartered to develop new products created high level feature diagrams called walking skeletons. They broke these features into epics, and then stories, which they sized and prioritized with the Product Owner.

Being a new Scrum Master. Throughout the transition, the Scrum Masters realized that although they were responsible for making sure teams adhered to Scrum principles and to the process, they only had the same training and experience as the others who attended the training with us. What we had learned about adopting Scrum one team at a time didn’t apply. So we scheduled planning meetings and set up daily Scrums, referring frequently to our notes from class for what came next. We struggled to answer questions on the rationale behind Scrum activities. It felt like directing improv theatre—everyone had a broad understanding of what to do, what role to play and what the ending should be. How each team got to the end was different every time, with unexpected plot twists and turns. What promised to be a short planning meeting to nail down a few details turned into an hour long discussion ending with unresolved issues. In some sprints, everything went as planned; in others, there was a lot to talk about during the retrospectives.

V. WHY IT WORKED

A combination of good timing and good planning made wholesale adoption of Scrum the right decision for us. Here are some factors to consider if you are adopting this approach.

Experienced workforce receptive to change. Due to their experience and long tenure at Unisys, most of the engineers and the management team in the US locations had worked with one another over the years. They had established reputations and networks of expertise that made it easier to introduce change of this magnitude. It would have taken much longer and been more difficult to get results if management and the
teams had needed to build new working relationships. After years of uncertainty and company restructuring, engineers were ready to try something new. Scrum was introduced at a time when even those who were traditionally skeptical of wholesale organizational endeavors were willing to give the new processes a chance.

**Starting fresh.** The conventional wisdom is to make small changes over time, or to prototype first then scale. In our case, changing everything all at once gave everyone the permission, the encouragement, and essentially the dictate to change. Because the new direction was not negotiable, it eliminated debate about whether implementing Scrum was a good idea or not. Instead of trying to fit Scrum into an existing organizational structure, Hodapp created a new one designed to support and facilitate adoption of Scrum. Unlike the rest of the employees who had long histories together, Hodapp was new to Unisys and new to the organization. Without the baggage of our past, he resisted engaging in our territorial battles or dysfunctional cultural patterns and coached us in building new ways to relate with Scrum at the foundation.

**Coaching.** As an experienced Scrum Master, Hodapp was able to coach teams and individuals when things didn’t go smoothly. If he ever doubted our progress or abilities, he kept it to himself. His feedback, while frequent, was not judgmental. Perfect Scrum adoption would never be the goal. Instead he emphasized we should focus our efforts on getting better, learning, and understanding through doing and frequent retrospectives.

**Luxury of time.** Hodapp had been through the process of transitioning to Scrum before. He made sure that upper management knew that it would take the better part of a year to be fully functioning. Although the teams had product deadlines to meet, commitments were made carefully so that deliveries could be made without abandoning Scrum during the inevitable crises.

**Fix the lines of authority.** The wisdom is that Scrum won’t fix your problems, but it will likely expose your flaws. In our case, restructuring had left many groups top heavy with managers and leads who were accustomed to being in charge. Imposing Scrum on the organization with the roles of pigs (people delivering product) and chickens (interested in the outcomes) was a constructive distinction that set up new lines of accountability. The separation of powers built into the Scrum philosophy clarified who got to decide what and in what situations. In a culture where almost everyone had felt they were obligated to add value to every discussion, the chicken/pig dichotomy made it easier for everyone to stay in role and made meetings more productive.

**Visibility.** Management’s commitment to complete transparency of our work was critical to our success. Thanks to the public Scrum tracking tool, everyone could see which stories were in progress and what stories were in the backlog for each product release. Since our Scrum meeting dial-in information was published on a common calendar on the Cloud Engineering portal, anyone from engineering, management, marketing, and other stakeholders, could attend any Scrum, any planning meeting, and any review and demo. This openness helped dispel concerns and suspicions about our new, strange-sounding processes. In practice, few visitors attend, but everyone is aware of the option. Taking visibility one step further, after a few months, we began recording every sprint review and posting the files on the portal so those whose schedules didn’t permit live attendance could see sprint demos at their convenience.

**Coordination.** Instead of each team selecting a Scrum Master from the engineers on the team, management decided to create designated Scrum Master positions in each US development center. In addition to serving as Scrum Master for multiple teams, these Scrum Masters had the job of supporting the transition to Scrum well past the initial launch. Hodapp intended it to work like a classroom lesson plan: introduce concepts first and then take the understanding deeper as everyone gained more experience. The designated Scrum Masters met daily at first, then twice a week. These meetings helped keep implementation consistent across the organization and expanded the Scrum Masters’ understanding of what teams needed to transition effectively. As an additional touch point, two Scrum Masters shared responsibility for the same team for a few months to learn from one another on a daily basis. Twice a week, the full-time Scrum Masters met with part-time Scrum Masters in an open discussion to share concerns and best practices and to provide mutual support.

Out of these meetings came ideas for the Scrum Topic of the Month where we agreed to encourage the teams to focus on specific improvements. For example, one month we would encourage holding more effective daily Scrums and the next month ask Product Owners to hold more frequent grooming sessions. We also shared our own experiences with templates for agendas for planning meetings and retrospectives, the use of Scrum of Scrums, and story breakdown techniques, to name a few. With so many teams going through similar transitions at the same time, having the Scrum Masters meet regularly to coordinate efforts across teams made learning proceed more quickly than if we had approached Scrum one team at a time.

**Commitment.** Once the organization was formed, the roles assigned, and sprints were underway, there was no turning back. Everyone knew this wasn’t an experiment. Through Hodapp’s monthly All Hands meetings, blog posts, and frequent on-site visits, he reinforced the message that he was absolutely committed to helping us make Scrum work. Although he acknowledged that management was making a big investment and there were risks, he had seen it work, he knew it could work, and he had confidence in us to make it work. So what happens when the VP who initiated Scrum rearranges his schedule for six weeks to dial in to every sprint demo? After everyone got over the stress of the first few demos, the engineers realized that Hodapp was their biggest cheerleader. And if we weren’t convinced of his conviction before, we were convinced then. Team by team, Hodapp reinforced with his presence that Scrum was important and his commitment was unvarying.

**VI. LESSONS LEARNED**

While we are very satisfied with the progress we’ve made and the results we’ve had adopting Scrum all at once, there are
several changes we would have made that you might consider if you are attempting this method of rollout.

**Formal training for everyone.** While training the entire management staff provided a critical mass of people who understood what we were trying to accomplish, having new Scrum Masters give an overview of the mechanics of Scrum to the rest of the organization left gaps in understanding. Though it would have been an additional expense, holding a formal class, even a half-day webinar, taught by a certified Scrum trainer would have given attendees the same baseline of knowledge. Engineers questioning the rationale behind Scrum concepts would have been able to get answers from someone with a depth of knowledge and experience right at the start. Since not everyone can attend a class and new people will join teams, it’s always important to have training available to bring others up to speed.

**More emphasis on story grooming.** Writing and grooming user stories seems like a simple process during Scrum training. When many teams start sprinting at once, no one starts off as an expert at writing stories. In the mad rush to convert requirements to stories, many stories had the correct format, but were often too big, too ambiguous, or too prescriptive. Product owners and teams who held grooming sessions once or twice a week over time refined their backlogs so that planning meetings were short, and there was no wasted time between sprints trying to figure out what to do next. For one untended product backlog, we instituted “speed grooming” where we spent a maximum of five minutes discussing a story to categorize it as “ready for a team,” “obsolete,” “more grooming,” etc., so the stories could be readied for upcoming sprints. Most of our stories were for system software, not end user features, so teams consistently underestimated how long a story would take, and would complain that it couldn’t be broken down further. All of sudden they were being required to break down work that used to take months into tasks of a few hours. In hindsight, we could have reduced stress on everyone if we had devoted more time upfront to developing everyone’s skills in writing stories and decomposing epics.

**Keep teams stable.** When we launched multiple Scrum teams, we tried to keep existing teams together and form new teams based on skills. As we gained experience, we realized that there were skills shortages in certain types of expertise that wasn’t possible to develop in a sprint or two. There was pressure to move engineers with certain skills to the teams that needed them for work on upcoming sprints or to address critical customer requirements. When moves occurred without much notice, teams couldn’t plan effectively if they didn’t know who would stay on a team. After several months of frequent changes, product owners and people managers agreed to minimize team changes, and even talked of instituting a “trading deadline.”

**More patience.** Instituting so much change at once can lead to frustration. Everyone is learning and adjusting at a different pace. Teams will embrace Scrum to different degrees and on different timetables. Under stress, people will fall back into old patterns of taking charge and telling people what to do. Hodapp frequently coached us that the goal is process, not perfection. He confirmed that it was acceptable to fail, to have unusual looking burn down charts, or to leave stories unfinished if that’s what took place. It was better to reflect reality than to “cook the books” to make things look right.

**Know when to be flexible.** When you implement Scrum from the top down, across the organization all at once, it is likely that you will need compromise on some Scrum tenets at first. You have to accept your start state, and don’t have the luxury of starting small and building up over time. While we compromised initially on some aspects of Scrum that were too difficult to accomplish given our established engineering practices, Scrum Masters ensured that all teams practiced the core pieces of Scrum, including planning, retrospectives, reviews, and daily stand ups. The daily standup was an easy concept for everyone to grasp, and it served as a daily reminder that our way of doing business had changed. We also insisted on two-week sprints. Cycling through planning, sprint, demo, and retrospective every two weeks gave us twice as much practice at the process than if we had relented to pressure from engineers for four-week sprints. Having retrospectives every two weeks forced the teams to reflect on what worked well and what could be improved, helping the teams more quickly grasp the power of self-organization.

Scrum preaches test driven design and releasable product every two to four weeks. For some of our newer products that had been rushed to market, there were few tests and, due to restructuring, no test teams. Fortunately, engineers were used to testing each others’ code within the team. But there were no automated builds, and no automated tests, and some test cycles took an entire week or more. Since multiple teams were in the same situation, it was easier to get buy in to launch a Scrum project focused on creating an organization-wide structure for automating builds and testing that all Cloud Engineering teams could use.

Because team membership was in flux and story size and completion varied greatly in each team, we have deferred calculating velocity. While teams size stories, and product owners and Scrum Masters look at story points per sprint, we don’t yet hold the teams accountable for maintaining or improving velocity. We expect to phase in tracking velocity as teams stabilize and as they get better at breaking down stories into small increments. Because we’ve seen so many other benefits from using Scrum such as visibility of work and clear delineation of roles, it hasn’t been a high priority to prove that we are delivering software faster.

**Allow time for attitude adjustment.** For some people, the sudden switch to Scrum was a difficult adjustment to make. Former managers, team leads, and project managers suddenly found themselves as members of self-organizing teams pitching in and being expected to deliver tangible results. They had to resist the tendency to take charge and tell people what to do, especially when things went wrong. To accommodate the need for day-to-day expert advice to guide team members, we designated Subject Matter Experts (SMEs) who work both as active team members performing tasks on committed stories and also serve as consultants to one or more teams in their areas of expertise. Scrum Masters coached the SMEs to step back, to ask questions, and to let the team come to an agreement about how to approach a problem. Again, by
implementing Scrum everywhere in the organization, it made it more acceptable for these transitions to take place since so many former team leads were affected in the same way.

**Watch for caged birds.** A maverick team that discovers Scrum on its own can relish the freedom and responsibility that self-organizing brings. When Scrum is imposed on many teams at once with decades of experience in hierarchical culture, the dictate to self-organize can be paralyzing. Engineers may have worked for decades with managers telling them what to do. A Scrum Master’s question, “How does the team want to handle this?” can result in a very long, painful silence. Patience, coaching, and encouraging participation from everyone can coax a team toward self-organization.

**Everyone needs support.** Introducing change of this magnitude creates stress and anxiety. “Birds of a feather” meetings to share concerns and work through issues can be helpful. The Scrum Masters met regularly to coordinate their work. The people managers met weekly to deal with staffing issues. The development teams had regular retrospectives to address their concerns. While the Product Owners met to discuss technical issues, they didn’t have a forum to discuss common issues and resolve problems. In retrospect, encouraging Product Owners to form a similar group for mutual support could have averted communications disconnects that later became apparent.

**Manage drive-bys.** In our training we learned the concept of drive-bys: work requested that is outside of the committed stories for a sprint. When creating a new Scrum organization with current engineering resources, the existing commitments and demands for expertise don’t just vanish; they must be managed. Almost everyone in our organization had specialized product expertise needed by non-Scrum departments that was available nowhere else. Add to that a culture of dropping everything to respond to any customer request, and you set the stage for burnout and incomplete stories. Although we accounted for this need by allocating 15 percent of everyone’s time to drive-bys and overhead and committing only 85 percent of their time to sprints, some in-demand engineers’ time was overcommitted. Eventually we realized that engineers needed guidance on how to manage requests from both inside and outside the organization, and the permission to say no, next week, or contact my Scrum Master or People Manager.

**Judicious use of spikes.** Imposing Scrum on existing projects exposed gaps in designs and testing that in many cases would require more than one planning meeting to address. By taking advantage of holiday weeks and other scheduling opportunities, we set up one-week or partial week spikes, complete with stories and estimates, to catch up on designs and research. As the teams got more comfortable with the idea of spikes, they created design stories within sprints, and design tasks within stories to address these design and architectural gaps as they were encountered.

**Be prepared for speed.** By marshalling the resources of so many people at once in the Scrum framework, we were able to make quick progress on many fronts simultaneously. For example, in a two-week spike, a small team with one US architect and two Bangalore engineers designed the architecture for a new interface. At the demo, management was so impressed with the results they gave the green light to proceed before the lead architect was even aware of the project. One team was scope out a new product area and within a few sprints had demonstrated that their design would work. I asked, “Where do you think you’d be if we were doing this project without Scrum?” An engineer replied, “We’d still be arguing about the functional design.” Success with whole organization implementation of Scrum gets attention. Our Bangalore development teams had adopted the Rational Unified Process (RUP) for their methodology, and when the Cloud Engineering organization was created they were given the option to stay with it. After Hodapp’s first trip to Bangalore and our initial results, the Bangalore teams transitioned from RUP to Scrum for full integration within the organization.

**VII. Summary**

If you are considering wholesale implementation of Scrum, consider the following:

- Don’t need upper management to be Scrum Masters, but it might not hurt. Having upper management engaged, supportive and visible is critical for wholesale organization involvement with Scrum. They need to be in training sessions and participating in their process roles as invested stakeholders.
- Find a rallying point. You might not have a recession as a spur to receptivity, but a common, critical cause to motivate change will make it easier to explain how Scrum can contribute to the success of that cause.
- Give the teams and individuals time to learn, experiment and practice. Don’t be in a hurry for improvements and exceptional results. Process and culture shifts of this magnitude take time.
- Don’t expect perfection. Steady progress is better than engineers obsessing about perfectly sloped burn down charts.
- Don’t economize on training. Everyone needs to understand the concepts and how to apply them in as much depth as possible.

One final thing that we’ve learned is that with Scrum, learning never ends. There are no completion criteria, and no definition of “done” applies. No matter how many groans I get after every retrospective, I still keep saying, “We’re still learning” and probably always will. And so will you.

**Acknowledgment**

Many thanks to Mark Hodapp, Bob Ohren, Dave Meyers, and especially Rhea Stadick for comments on drafts of this paper.