Formalizing Agility, Part 2: How an Agile Organization Embraced the CMMI

Steven W. Baker
Software Engineering Methodologist, DTE Energy
bakersw@dteenergy.com

Abstract

Most large IT organizations need the best of both worlds – working software that supports ever-changing business needs, and a process for delivering software that is predictable, trainable, and auditable. Organizations with agile software teams often receive timely, cost-effective solutions of sufficient quality. Organizations with formal processes often benefit from industry-recognized certifications and robust process improvement mechanisms. Rarely does a single, large IT organization achieve both.

DTE Energy has such a combination: its lightweight yet CMMI-compatible methodology is now used within its 600+ person IT organization to deliver and support working software. Its small teams embrace core agile principles as they provide “just enough” solutions that satisfy maturing business needs. Yet they passed two SCAMPI appraisals toward formal CMMI Level III accreditation, scheduled for mid-2006.

This report extends the Agile 2005 experience report on DTE Energy’s agile IT organization’s journey toward CMMI staged maturity accreditation. This report briefly recaps their seven years of agile experience, presents governance mechanisms and change management techniques, and highlights their four-release, one-year plan for CMMI Level III accreditation. Finally, this report offers suggestions on embracing a formal process framework that are applicable to any organization.

1. Context and History

DTE Energy, a Fortune 300 company headquartered in Detroit, Michigan, USA, is a diversified energy company involved in the development and management of energy-related businesses and services worldwide. DTE Energy’s Information Technology Services (ITS) Organization provides leadership, oversight, delivery, and support on all aspects of information technology (IT) across the enterprise on behalf of its 11,000 employees.

One of the company’s strategic IT goals is to produce, deliver, and support world-class software. Our software engineering (SE) organization of over 350 professionals includes development, test, business analysis, project management, and governance groups.

In 1998 a part of the SE organization began using what was later called agile methods to enable short-cycle, iterative and incremental delivery of working software. While they were successful in meeting the needs of the business, the IT organization continued to seek ways to benchmark and improve not only our software but the processes we employed to deliver and support it.

Another IT group embarked on a program to learn about and benefit from the Capability Maturity Model (CMM). From 1999 to 2001 the two made progress in formally defining processes and in writing software using agile techniques. While each provided value alone, we needed to reconcile and unite the two approaches.

Our IT leaders commissioned a Software Engineering Process Group (SEPG), chartered to oversee the process improvements. The SEPG brought together practitioners from software engineering and business-facing groups; the author served as methodologist and facilitator.

Leveraging the continued success of our “house blend” of agile methods for software delivery, the SEPG agreed to use our same agile methods to deploy and support a software engineering process framework.

Like any product, we needed a set of requirements and a suite of acceptance tests. We selected the Capability Maturity Model Integration (CMMI) from the Software Engineering Institute (SEI) [1]. The CMMI provided us with a set of features (“what” our product should do).

The SEI also provided a ready-to-use acceptance test framework - the Standard CMMI Appraisal Method for Process Improvement (SCAMPI) [2]. It gave us a way to validate and verify that we simply document what we do, do what we document, and prove it.

Rather than following a waterfall “big bang” approach we planned to embrace our own agile methods. In effect we would use our agile software process to produce, test, deploy, and maintain our agile software process [3].
2. Commissioning the Program: Release 0

In July 2005 we launched our process program like we would a software project - with a timeboxed Release 0. The focus of Release 0 was to plan enough to get started and to ensure the team was ready, willing, and able to proceed. We began by addressing some key questions.

First, we needed to clarify and confirm our intent and rationale to proceed. What was the case for change? Weren’t we already delivering software of sufficient quality? Weren’t things going “well enough” so far? To extend an eXtreme Programming (XP) practice, we needed a system metaphor to rally around [4].

Continuous improvement is core to our company. Believing in “we can do better” over “if it ain’t broke…” we sought to improve quality, reduce costs, and increase satisfaction. For us the CMMI was a means to an end rather than a final goal [5]. Yet we asserted that external validation and verification of our progress was critical.

Second, we needed to tackle the “buy vs. build” decision. In this context, would we hire a company to come in and “do the CMMI” to us, or would we roll up our sleeves and embrace the CMMI ourselves?

Our prior experience in leveraging the CMM taught us many things. Our earlier program had full-time resources designing and deploying processes to the practitioners. The processes, templates, and other work products were well-designed and comprehensive, yet our practitioners struggled to fully embrace them.

We learned that effective change comes from within, and that process ownership is more fully ingrained when the practitioners are actively involved in its definition and refinement. We agreed to take the “build” path and sought internal practitioners on a voluntary and part-time basis (above and beyond their “day jobs”).

Third, we needed guidance to navigate the waters and to clarify the model within our context. We sought and retained an experienced consultant, who understood both agile methods and the CMMI, to guide us along the way.

Fourth, we needed to clarify roles and responsibilities that were new to many of us. Process Owners stewarded the “care and feeding” of one or more process areas with an enduring, cross-functional Special Interest Group (SIG) to design, deploy, and manage the processes.

We also formed a small, internal appraisal team to validate and verify our SE organization’s process area performance by using the SCAMPI model.

To provide CMMI governance we formed a small, cross-organizational senior steering committee. They supplied the program with CMMI-related requirements, guidance, decision-making, and organizational support.

These various teams were staffed from within our IT organization, recruiting those who had passion, ability, and capacity to make and honor project commitments. This was especially challenging as we all had full-time responsibilities to our business partners.

Our decision to do this to ourselves encouraged this approach. We planned to pilot our new process assets as soon as possible, see what helped us and what hindered us, and adjust accordingly (“feedback early and often”).

Several of our business partners quickly embraced our program, recognizing the potential value to their current projects and to the organization. Their advocacy proved to be of significant value during crunch times.

In Release 0 we also planned the program timing and structure of Release 1 that included a program timeline, requirements backlogs, metrics, team capacities, and testing, training, communication plans.

A key source of requirements was an informal Class C pre-assessment from the spring of 2005. This tested our as-is product (our existing processes) against the set of agreed-upon requirements (the CMMI framework). The brief, low-ceremony appraisal provided a documented set of insights into our process strengths and weaknesses.

Many software products require several releases to be sufficiently feature-rich, with each release incrementally and iteratively delivering additional value. This proved to be the case for us as well: our roadmap called for three releases (see Figure 1, below), each with acceptance tests of increasing rigor as the set of features expanded:

- **Release 1** - Focus on ten Level III process areas targeting the CMMI staged representation, and test ourselves with one informal Class B appraisal.

- **Release 2** - Focus on six Level II process areas plus prioritized Release 1 defects, and test ourselves with one informal Class B appraisal.

- **Release 3** - Focus on sixteen Level II and Level III process areas plus prioritized Release 2 defects, and test ourselves with two formal Class A appraisals.

Our case for change was articulated, our approach was clarified, and our initial scope was proposed. With a team and a test harness in place, we completed Release 0 with confidence and with IT management’s “green light” to proceed with Release 1.

---

**Figure 1. Program Timeline**

<table>
<thead>
<tr>
<th>Release</th>
<th>Two Iterations</th>
<th>Seven Iterations plus a Warranty Period (WP)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Release 0</td>
<td>July 2005</td>
<td>August – December 2005</td>
</tr>
<tr>
<td>Release 1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Release 2</td>
<td>January – March 2006</td>
<td></td>
</tr>
<tr>
<td>Release 3</td>
<td>April – June 2006</td>
<td></td>
</tr>
</tbody>
</table>
3. Solution Delivery Process: Release 1

Our plan for Release 1 called for several mini-projects under one program umbrella. Each small project team was a Special Interest Group (SIG) that focused on one or more process areas from the CMMI model. Each SIG had a Process Owner (the voice of the customer), SIG Coordinator (the project manager), and subject matter experts from the software engineering community.

Four of our SIGs covered more than one process area:
- **Software Engineering** - Validation, Technical Solution, and Verification.
- **Project Management** - Project Planning, Project Monitoring and Control, Risk Management, and Product Integration.
- **Requirements** - Requirements Development and Requirements Management.
- **Organizational Process** - Organizational Process Definition and Organizational Process Focus.

Three of our SIGs covered their namesake process area: Configuration Management; Process and Product Quality Assurance; and Measurement and Analysis.

The SEPG, being a governing SIG in its own right, covered Organizational Training and Decision Analysis and Resolution. We denoted the remaining process areas as being out of scope for our Organizational Unit.

As an agile organization, we believed in traveling light with only necessary and sufficient documentation. As we looked at the CMMI model, it sure felt document-heavy! How were we going to remain true to our agile principles with what could be hundreds of new artifacts?

One of our important realizations was that the CMMI as a model, and the SCAMPI as a test harness, does not explicitly prescribe or require any particular work products or artifacts. Rather, it simply proposes and seeks evidence that processes and practices were performed. It was in our hands to bridge this gap.

In our corporate environment we recognized the need for “just enough” formality. With hundreds of resources working on dozens of projects, we sought an appropriate level of predictability and repeatability. We chose to rally around our reasonably well-described agile processes that encourage individual ingenuity, business partner creativity, collaboration and teamwork, and the use of reusable process asset components.

Defining these process areas was not enough; the work products were only valuable if the community knew about them and how to meaningfully apply them on their projects. We realized that process definition was only one-third of the challenge: our organization would also require effective communication and training materials to fully realize sustainable process improvement.

3.1 Synchronizing the SIGs

Each SIG was chartered to codify the current state of their process area(s) with simple process flows, inventory their existing work products, identify the gaps using the CMMI model as a reference, and propose a minimal amount of new work products to bridge those gaps.

We launched each SIG as it was ready. The SIGs followed the same timeboxed schedule but were slightly out of phase with one another (often by only a week or two). At first this was not of concern as early deliverables included a manifest of current-state work products and a depiction and description of current-state processes.

Communication challenges between the SIGs soon emerged as dependencies between the process areas were uncovered. Process flows and descriptions accurately reflected our current state, but were inconsistently articulated. Process deliverables identified by a SIG were sometimes a challenge to reconcile with other SIGs given the schedule (earlier or later) and the assumptions made.

Our first iteration was relatively successful, but in Iteration 2 as we planned for Iteration 3, we decided to learn from our experience and adjust our plan (in keeping with our “feedback early and often” and “fail fast” principles). We extended the timeboxes of some of the SIGs so that we all ended Iteration 2 at the same time to synchronize our schedules going forward.

To kick-off Iteration 3, we held a two-day offsite with all SIG members. The purpose was to communicate and validate each process area, to construct a “big picture” of all process areas and interdependencies, and to propose an integrated training and communication approach.

On the first day, each SIG presented their work with plotter-size process flows using sticky notes and markers. This exchange of information was enlightening as we clarified assumptions, identified and negotiated interdependencies, and agreed on common definitions.

The morning of the second day saw our SIGs working together on an integration exercise. We produced an “uber-flow” depicting roles (down the side) and process steps (from left to right) from all of our process areas. We had a soup-to-nuts picture of our full process.

We completed the afternoon of the second day with a facilitated exercise to propose and agree to our training and communication strategy. We created a grid with process roles down the side, process areas across the top, and used a numbering system to reflect the depth of knowledge required for a given role of a given process.

The event had a celebratory feel as we recognized and appreciated the results we had achieved thus far. Our SIGs were proud of the products they produced, and supported one another in moving forward together.
3.2 Approaching the Class B Appraisal

The SCAMPI calls for boundaries to clarify what parts of the company are in-scope. Embracing our IT strategic plan, we defined our “Organizational Unit” as agile-based software projects with in-house resources using J2EE technology to deploy internal web-based applications into production before the end of 2005.

By Iteration 3 our work products were “done enough” to begin piloting them on software projects. Our intent was to (a) validate and verify that our process flows and artifact templates were of value, encouraging a timely feedback loop to the SIGs; and (b) prepare for the upcoming SCAMPI Class B appraisal, testing the maturity of our Organizational Unit.

We initiated five pilot teams and paired them with liaisons from the various SIGs. Work was curiously challenging at first since the ink was not yet dry on our process products. Culturally, we were comfortable with dynamic requirements, daily builds, and incremental software growth, yet it was quite another story trying to leverage fluid process models and project templates.

In parallel, the pilot projects began preparations with our in-house appraisal team. Our consultant served as Lead Appraiser, and our team of eight included project managers, SIG members, and governance leads. Their diverse experiences, roles, and personalities ensured that our appraisal would be objective and fact-based.

Our onsite consultant wore additional hats as we learned more about the model and its implications. He provided in-house training on the CMMI model to the SIGs, the appraisal team, the pilot projects, the SEPG, and other IT management and stakeholders.

In turn, he appreciated our passion and experience with applied agile methods in a corporate environment. Rather than rigidly struggling to repeat what had worked at other companies, he embraced our desire to use our processes to improve our processes. We sought to do this to ourselves, and he proved to be an excellent accelerator.

By the fall of 2005 we were ready to begin our first Class B appraisal. While our process scope for Release 1 covered both Level II and Level III areas, our appraisal primarily targeted Level III because (a) we wanted to test our target of Level III early and often, and (b) we needed more than one release to sufficiently embody some core Level II areas (like CM and MA) in our organization.

The appraisal was to examine a “second order” effect: the pilot teams were to read, understand, and apply our standards, guidelines, and lightweight templates; the appraisal team was to assess our pilot teams’ process performance (rather than to audit their work products). They communicated the results during our retrospective.

3.3 The Release 1 Retrospective

In December 2005 we held an offsite retrospective to review Release 1 and preview Release 2. Attendees included the SIGs, the appraisal team, the SEPG, the pilot projects, and management stakeholders. We focused on three key themes: the project retrospective in the morning; the appraisal results and celebration lunch, and preparation exercises for Release 2 in the afternoon.

We allocated two hours for the retrospective so the entire group had sufficient time to engage in meaningful dialog. The facilitated session posed four open questions:

- **What Went Well** - We produced our process products quickly; we provided value to the pilot project teams; we embraced our agile principles; and our project gave the organization greater focus.

- **What Did We Learn** - We learned our process as we documented it; how we support our strategic goals and leverage industry frameworks; and we have clarity into what we know and what we don’t know.

- **What Should We Do Differently** - We should set bigger goals, involve more of the population earlier, communicate specific benefits sooner, and better integrate our processes to reduce inadvertent overlaps.

- **What Still Puzzles Us** - How to better integrate new SMEs into the SIGs; how to more accurately predict the level of effort required to use the process; and how to flatten the layers of hierarchy in decision-making.

Headlining the agenda was the informal Class B appraisal results and findings. Having followed a parallel, four-iteration Release schedule, the appraisal team had gathered and analyzed 600+ data points across 67 specific practices. Did we pass our acceptance tests?

Indeed, we had made significant progress in each area from our midsummer Class C pre-assessment. Especially given the more stringent Class B requirements, our “Largely Implemented (LI)” ratings were well-earned. Two process areas that had less attention (OT and DAR) were appropriately assessed “Partially Implemented (PI),” providing feedback and scope for Release 2.

With Release 1, we deployed updated role-based standards and guidelines, software engineering process flows and definitions integrating our best-practices with the CMMI framework, orientation and training materials on our process and work products, and a refreshed web-based process asset library (PAL) containing our suite of process products including templates and examples.

To conclude the workshop we discussed the schedule, structure, and proposed scope for Release 2. We formed break-out groups to further examine our strengths, weaknesses, opportunities, and threats, and brainstormed ways to better enable success.
4. Solution Delivery Process: Release 2

We learned much during Release 1, not only about the CMMI and SCAMPI models but how to apply them in our agile culture. This was valuable as we refined our Release 2 proposal during the Release 1 warranty period. Our IT leadership team dedicated funding for two additional releases and they also committed funds for the SCAMPI appraisals in the spring and summer.

Our timeboxed Release 2 began in January 2006 with three, three-week iterations for development, testing, and pilots, with a “functional freeze” milestone (no new features). Next was a single three-week acceptance test iteration with a “code freeze” milestone (no changes) before the production build and one-week warranty period, planned for early April 2006.

Release 2 scope included:

- Iterative maturation of the “development code base” from user and SIG feedback, Class B appraisal results, and our feature backlog from Release 1.

- Incremental expansion of our process assets to provide a baseline for other project types in addition to our traditional 13-week agile approach; our backlog included maintenance, waterfall, COTS, and a six-week agile methodology we dubbed “Dash.”

- Acceptance testing by means of an informal Level II Class B SCAMPI appraisal, again leveraging our consultant, our in-house appraisal team, and a suite of appraisal projects.

- Prioritized defect resolution of the “production code base” (some defects were fixed only once in the development set of work products, while “critical” defects were patched in production as well).

Our process products were now in production, and we planned to support our users (the IT software engineering community) with communication, training, and feedback mechanisms. Rather than forming a separate production support group to support our users and resolve prioritized defects, we allocated capacity within our development team (the SIGs and the SEPG) to production support.

Process defects, including extraneous steps, misstated procedures, and cumbersome new templates, were logged and tracked with a web-based corrective action system. The appropriate process owner received, prioritized, and addressed the defects as scope to be negotiated and managed within the release.

We ramped up our communication and training efforts. One of our favorite principles is, “anything that takes time and costs money is scope.” We allocated capacity to develop and deliver a suite of communication and training assets, with the intent of providing the right messages to the right audiences at the right times.

4.1 Reality Strikes Back

Usually the first quarter of each year is interesting in many ways: new and confirmed corporate priorities, a new budget cycle, and a new suite of products to produce. Our plans were to leverage a selection of IT projects that aligned with our Organizational Unit for the informal Level II Class B appraisal as part of Release 2.

Yet this year was not unlike previous years; some projects indeed started in January, whereas others, for appropriate reasons, were deferred. We had inadvertently placed a critical path item on an external dependency: some of our Organizational Unit projects were either not in-flight or would not be far enough along.

Without a sufficient pool of projects, our assessment would not meaningfully represent our organization. By the end of Iteration 1 we were at a crossroad. Should we wait for a critical mass of Organizational Unit projects? Should we proceed with the project but defer the appraisal? Should we expand our Organizational Unit beyond traditional agile software engineering projects?

Enter the Decision Analysis and Resolution (DAR) process, a CMMI Level III maturity process area. We seized the opportunity to move the process ball forward (our appraisal results accurately reflected our partial maturity in this process area) while we addressed a real-world project issue (our Organizational Unit vs. Reality).

We piloted a lightweight DAR process and broadened our Organizational Unit beyond traditional agile projects. Our feature backlog included other software engineering methodologies still in-use; now the stakes were higher and our project team accepted the challenge.

To honor a project principle - to do the CMMI to ourselves rather than have someone else do it to us - we revisited the composition of our SIGs. With new methodologies now in-scope, our teams needed subject matter expertise to advocate and work from within.

We used (and in turn refined) our communication and training materials to bring the new SIG members up-to-speed on the project, the process, and the products. We appreciated the irony of agile practitioners advocating the value of a process having “just enough” rigor to be objectively applied, assessed, and improved.

Our SIGs continued to apply our agile methods as our scope expanded beyond them. We reviewed each process area and proposed tailoring guidelines based on our agile baseline. Rather than refining the process in a vacuum, we partnered with the appraisal projects to address their needs within the guidelines of the process model.

We found ourselves rallying around one of our new best friends in Release 2: Configuration Management, a CMMI Level II maturity area.
4.2 Configuring Configuration Management

Our IT organization has long been proficient in storing, tracking, and versioning its code work products (including configuration and build scripts). We had pockets of excellence in managing non-code work products (including plans, test cases, and defect lists) but “actual mileage varied” from one project to the next.

Our mid-2005 Class C pre-assessment surfaced our need to rally around Configuration Management, and we improved our process considerably during Release 1. With our Release 2 process scope having expanded beyond the traditional agile methodology, we found ourselves celebrating that progress.

A key tenant in agile methods is to focus on what we needed to produce rather than how we went about producing it. In other words, our agile plans addressed the desired deliverables (nouns) over the activities (verbs) we performed to create those deliverables.

Configuration Management (CM) plans clarified this distinction. A core element of the CM plan is a Work Product Matrix (WPM) listing the types of work products a team intends to produce during the course of the project. The team declared what categories of outputs they intend to deliver (code, test cases, defect lists, build scripts, etc.) to satisfy the needs of the business.

This was a powerful tool. Deliverables are scope (they take time and cost money) and clarity is essential. If the business wanted user manuals, we added it to the WPM and estimated and prioritized them accordingly. If the business did not need user manuals, the WPM reflected this decision as a baseline for project commitments.

We realized that the CM plan and the WPM were ideal for our Process and Product Quality Assurance (PPQA) team. By listing the deliverable types we intended to produce (code, test cases, use cases, etc.) the PPQA group had a foundation to assess a team’s compliance (produced work products were evidence that the corresponding processes were performed).

It was with this CM framework that we expanded our process assets beyond our traditional agile foundation. By tailoring the CM plan and its work product matrix, we declared what artifacts a methodology would produce. What distinguishes one methodology from another is the set of work products it delivers along the way.

Our appraisal and pilot projects learned about and soon adopted the CM process. During initial planning, each project team tailored the “default” CM plan. To learn from their insights (and to ensure they did not tailor themselves out of alignment with our process framework), we piloted a brief stakeholder review session to review and approve the tailored CM plans.

4.3 Release 2 Results

In Release 2 we piloted a training initiative to build a community of process maturity. Our intent was to craft training materials that were relevant to our practitioners’ project roles and also sufficiently engaging so that they formed a self-supporting community.

We trimmed our four-hour orientation from Release 1 to two hours, supplementing it with role-based “builder tracks” developed by the in-house SIGs to provide just-in-time training for key process roles (developer, tester, business analyst, and project manager).

Our new Methodology Manual brought together core principles, terminology, role descriptions, process flows and step descriptions, simple policy statements, tailoring guidelines, and governance mechanisms into a single handbook. It joined our streamlined templates and refreshed role-based standards and guidelines on our web-based process asset library.

By the end of the second iteration our appraisal project teams had begun using the process, providing feedback, and preparing for the appraisal. While we had a diverse Organizational Unit to appraise, each project team simply needed more time to produce their agreed-upon work products as they performed the process.

We exercised our lightweight Decision Analysis and Resolution (DAR) process to decouple the appraisal schedule from our project schedule. This was necessary but unfortunate, as we had separated the “user acceptance test” cycle from the release itself.

The informal Class B appraisal in Release 2 focused on Level II maturity areas with an eye on those Level III areas flagged in our Release 1 appraisal. Recognizing we needed to be “Fully Implemented (FI)” across the board by the summer of 2006, we encouraged this feedback to gauge our progress and provide opportunities to improve.

In late-April the in-house SCAMPI appraisal team conducted its readiness assessment, gathered almost 1900 pieces of evidence, conducted project team and functional area representative (FAR) interviews, accumulated the results, and presented their findings.

We and our IT leaders were proud of the results. Four of the six Level II process areas were “Fully Implemented (FI)” - Project Planning, Project Monitoring and Control, Configuration Management, and Measurement and Analysis. The other two were “Largely Implemented (LI)” - Requirements Management and PPQA.

The pull-ahead Level III areas were as we anticipated: Organizational Training was “Fully Implemented (FI)” and Decision Analysis and Resolution was “Partially Implemented (PI),” While our processes needed some work, we basically needed more cycle time to use them.
5. Solution Delivery Process: Release 3

In mid-April 2006 we launched the third release of our software engineering process improvement program. Similar to Release 1 and 2, we sought and received senior IT sponsorship of the project plan package. It contained a lightweight suite of release and iteration plans, team rosters, capacity plans, communication and training plans, appraisal plans, and our own CM plan.

Like Release 2, the schedule consisted of three, three-week iterations for incremental and iterative development and testing, a single three-week iteration for acceptance testing, a production build, and a warranty period. We began this release with a decoupled appraisal schedule that lagged by one iteration (the audit for Release 2 was still in-flight as we began Release 3).

We retained our SIG project team structure (by now each had stabilized into effective volunteer workforces with strong relationships with other SIGs, with the appraisal team, and with the in-flight projects) and our governance groups (the SEPG, our PPQA audit enablers, and our CMMI steering committee).

In Release 2 we cut two of our project types from scope due to pilot readiness issues. We did stabilize the Release 1 (traditional agile) process flows and piloted some for our maintenance projects. As we considered the process flows in more detail, we discussed the advantages of having full sets of diagrams for each methodology. Would increased clarity be worth the cost of maintaining multiple “flavors” of each flowchart?

For Release 3 we proposed “generic” process flows that would be methodology-neutral. We agreed to use our existing traditional agile process flows as a baseline, and sought to denote specific variations only where needed.

This led us to ponder what it really meant to codify a methodology in the first place. Reviewing our backlog of project types (waterfall, maintenance, COTS, and Dash) we assumed we would need to describe each with process flows, descriptions, work product templates, and the like.

Instead of expanding the amount of work products we needed to produce, we sought to maximize the amount of work not done (another key agile principle). We thus proposed to produce a reasonably well-defined “abstract” methodology, a set of clear tailoring guidelines, and a suite of configuration management (CM) plan patterns.

We agreed that each CM plan pattern would include a pre-tailored Work Product Matrix (WPM) for a given project type, listing the various types of work products a project team would produce. For example, one project type did not intend to produce JavaDoc since it was modifying legacy Oracle stored procedures; the JavaDoc work product was “tailored out” in that CM plan pattern.

We needed more experience with our lightweight Decision Analysis and Resolution (DAR) process. Our Release 3 plan called for a feature backlog of “DAR-able moments,” targeting five decision patterns. These patterns were to be reusable for important decisions like “buy vs. build,” third-party product and vendor selection, and managing significant change.

Leading up to Release 3 we focused on our solution delivery process itself, striving to embrace our process roles on day-to-day projects. Our intention was to understand the process before we automated it; our “applied maturity” approach was to walk before we run.

Since tools should be enablers to, not substitutes for, process, we directed little attention to tools that might accelerate our process. For example, our Configuration Management process called for non-code work products to be stored on a network share drive with consistent project directory hierarchies and filename taxonomies.

Without rocking the boat ahead of our CMMI Class A appraisals, in Release 3 we planned to explore potential tools that might automate certain aspects of our process such as requirements traceability, defect analytics, and non-code configuration management.

Release 3 called for expanded training in key areas. We planned to continue the role-based “process builder” training, having allocated capacity to refine and expand the materials based on feedback. We planned more focus to “train the trainers” so that additional members of our communities will be able to lead group discussions.

In addition to in-house training, we held CMMI and SCAMPI courses from the Software Engineering Institute. This was to expand our shared understanding of the models, and to ensure our appraisal team was fully certified to perform a formal Class A appraisal.

Last spring our informal Class C pre-assessment gave us the confidence and courage to set our sights on CMMI Level III accreditation. We had planned to combine the Level II and III areas in one Class A assessment.

This schedule posed several logistical risks which we decided to mitigate more deliberately. We refactored our schedule into two back-to-back Class A appraisals: first, a Level II and III accreditation. We had planned to combine the Level II and III areas in one Class A assessment.

As of this writing, the formal Class A appraisals are only two months away. Our software engineering groups are more confident than ever in our ability to apply agile principles across the organization. In a mere fifteen-month span, with part-time resources and a scant budget, our software engineering culture will have matured from delivering and supporting working software to providing agile-based, world-class applications at an industry-recognized caliber of process excellence.
6. Lessons Learned and Recommendations

Many IT organizations are relatively successful by delivering working software to the best of their ability. Others seek to continuously improve themselves, and desire a framework to leverage as a starting-point.

To those groups seeking process improvement, the following suggestions may provide some degree of value. Our large IT organization embraced agile methods and subsequently applied those same methods to our process improvement initiative.

First, clarify why you seek to improve your processes at all, and why you would leverage a framework like the CMMI. The case for change is important and a shared vision of the future will clarify each decision [6].

We sought the CMMI as a means to an end; our intention is to be a world-class IT organization delivering higher-quality software in a more timely and cost-effective manner. Rather than seeking formal CMMI accreditation as the final goal, we see it as validation that we are embracing a durable process framework.

Second, clarify who is doing the process improvements to your organization. Some companies hire a squad of consultants to bring in a new process; with much fanfare the expert consultants roll out a complex framework and then roll out of town, leaving the organization to sort it out for themselves.

We believe that sustainable culture change comes from within. Many teams seeking to embrace agile methods hire an agile coach to mentor them; likewise, we hired an experienced CMMI consultant to guide us through the model and its real-world application.

Third, clarify how you will go about developing, deploying, and maintaining process improvements. Some in the industry advocate a custom methodology with agile-techniques to accelerate process improvement.

We prefer fewer rulebooks over more, and generative ones over prescriptive ones. We believed that tried and true agile methods were well-suited to non-software products, and we subsequently demonstrated that agile methods supported and enabled our success.

Our organization leveraged an adaptive and time-boxed approach with small process-focused teams, and piloted our work products with a set of in-flight projects. With unwavering support from senior IT management, and our lightweight governance, metrics, and assessment groups, our passion for process improvement continues.

The CMMI actually encourages creativity; we simply viewed it as a framework rather than a cookbook. As such it provided “just enough” rigor as we formalized our agile methods to deliver and support world-class software to our business partners.

7. References


