Release and Team Utilization Planning Application

By
Jeff Brooks

A project submitted in partial fulfillment of the requirements for the degree of
Master of Science in
Computer Information Systems

at
Grand Valley State University
April, 2014
Table of Contents

Abstract .............................................................................................................................................. 3
Introduction ......................................................................................................................................... 3
Background and Related Work ........................................................................................................... 5
Program Requirements ...................................................................................................................... 8
Implementation ................................................................................................................................. 9
Results, Evaluation, and Reflection .................................................................................................. 11
Conclusions and Future Work .......................................................................................................... 16
Bibliography .................................................................................................................................... 18
Appendices ....................................................................................................................................... 19
Abstract

This project involves the development of a web-based application to manage release planning and team utilization. The goal of the project is to improve mid to long term planning for agile software development projects and managing the capacity of the teams working on them. While the tool and process can be used by any software development methodology that is using an approach involving several releases to complete the project, the primary focus of this project is based on the experiences and needs of an organization using an Agile/Iterative software development process. The system was developed using Oracle Application Express 4.2, Oracle’s RAD tool for developing Web applications with SQL and PL/SQL.

Introduction

The motivation for the project is based on the experiences of a large health insurance company with several affiliated companies that implemented an Agile/Iterative software development process about 6 years ago. The Michigan office provides all of the claims processing and IT development and support for its Michigan business as well as its affiliates located in Indiana, Kentucky, New Mexico, North Carolina, Ohio, and Tennessee. While the Agile/Iterative approach resulted in faster turnaround time for delivering working software, it also presented challenges with providing the business with a roadmap of completion target dates of new features on larger projects, the overall project target completion date, the capacity of the development teams, and when new projects could begin based on team capacity. The goal of this project is to attempt to overcome at least some of those challenges.

One of the core Agile principles is to deliver working software frequently e.g. 2-4 weeks and for a business owner to work with the team to regularly “groom” the project’s backlog to ensure the highest value features or user stories are being prioritized and worked on each iteration. With the team’s focus on the current iteration, it is easy to lose sight of the longer term plan. For the purposes of this paper, the terms “feature” and “user story” will be used, both being related to defining a user’s requirement with the term “feature” being a higher level requirement that is generally comprised of a group of related user stories which are more detailed requirements. With the primary emphasis on delivering working software each release and working on multiple projects simultaneously, the teams became accustomed to focusing on creating and planning their user story cards for the current and next releases. This focus led to avoiding the building and estimating of a full project backlog and any planning for feature releases.

This particular organization’s iteration and release follow a consistent 4-week iteration/release timebox. Since the company often had new projects being introduced each release and there was a lot of context switching for the teams working on multiple projects of varying length (2 to 15 months), they would spend little or no time on the mid- to long-term planning at the feature level. They believed such longer-term
planning offered little value since the plan would likely change month-to-month as new project priorities were introduced. This left the Project Management Office (PMO) with a big problem: there wasn’t enough data to show projected project completions and provide management with a picture on the capacity of the teams. The PMO therefore struggled to provide estimates of when new projects could begin.

To manage the story cards during the current release and plan the next release, the teams used an Agile Project Management tool called Mingle from Thoughtworks. While they were able to use the tool to successfully manage their work at the user story level for both project and maintenance type work for the releases, the tool was difficult to set up for managing the feature-to-story relationships and for future release planning. Since teams were adding the user stories and estimates just before each release instead of building the backlog upfront, the burn-down charts produced by Mingle for most projects were ineffective. In Figure 1, the bars of Complete + Backlog continue to grow each iteration as the story cards are added each release versus building a backlog upfront. A proper, effective planning approach would produce burn-down charts where the combined bars would remain roughly level or only slightly increase and the ratio of backlog to complete stories would decrease as time progressed.

![Figure 1 - Example of Ineffective Burn-down Chart (story points)](image-url)
Without a project roadmap and a proper burn-down chart, the PMO and business leaders had difficulty planning the portfolio and other tasks that were dependent on the completion of software development. As a result, on larger projects, the Project Management Office (PMO) would frequently be required to submit a change request for extending the project end date and potentially additional budget. While the business was pleased they were getting working software early and throughout the development process, frequent project priority changes created a constant shuffling of work for the Development Managers trying to satisfy a number of active projects without having a good projection of when each would be complete. The business users would often be frustrated not knowing the roadmap for their project. Also, without a summary of the team’s utilization across several projects and their current and planned capacity, the PMO found it difficult to determine which teams needed additional resources, which teams might have had available resources to help over-allocated teams, where the potential bottlenecks were, and when new projects should begin.

Management needed visibility into project and portfolio level planning. The goal of this project is to develop an application that can provide that visibility. This project includes an application that can provide planning information to the PMO, the development teams and the business to make better data driven decisions in terms of the projected completion date based on the trend of completed and remaining work, the trend of the budget, and the team utilization showing their allocation across several projects.

Background and Related Work

These medium- and long-term planning problems are a common occurrence in Agile organizations. Hubert Smits (2010) has developed a five-level agile planning framework (Figure 2) to help organizations understand overcome these problems.

![Figure 2 - Five levels of Agile Planning](image-url)
In Smits’ agile planning framework, the product vision describes the future plans of the product features and what may differentiate it from its competitors. The roadmap represents a complete prioritized product backlog of features based on business priorities that defines how the vision will be realized. For shorter projects with only a few iterations, iteration planning may be sufficient, but longer term projects benefit from a more complete release plan and roadmap and regardless of the size of the project, there should be an overall release plan based on the prioritized portfolio.

One of the reasons the company cited in this paper struggles with medium- and long-term planning is a lack of a named product owner or a clear business owner that can provide the guidance required to complete this level of planning. The result is planning being defined by various projects that come in from a variety of sources that are not always prioritized based on any long-term concerns. A common example of this problem occurs when projects are worked on in response to a recent request from a demanding project sponsor which may not always be the strategically next-in-line project. Without a product owner or a steering committee that can provide a more defined product vision and roadmap, it is more likely the teams will experience constantly changing priorities and have difficulty with their medium- to long-term planning since there isn’t a consistent direction provided. Although the teams are not provided with a product vision and roadmap, the Release and Team Capacity Planning application developed in this project can provide the PMO and teams with projected end dates based on the slope of the burn-down chart which shows the remaining hours for the project by calculating the difference of the total scheduled hours and the actual hours, and the data needed to better forecast when they will have capacity available to help other teams or start other projects.

Another more comprehensive framework that provides guidance on scaling Agile for larger enterprises is the Scaled Agile Framework (see Figure 3) (Dean Leffingwell, 2014). Similar to the 5 Levels of Agile Planning, it defines different levels of planning an organization needs to consider including long-term planning at the Portfolio level, medium-term planning at the Program level, and iteration/sprint planning at the Team Level.

The company in this paper has done a decent job implementing Agile at the team level, but to progress to the next level in Dean Leffingwell’s Scaled Agile Framework, the company will need to focus more effort at the Program and Portfolio levels.
In regards to tools that can be used for the different levels of planning, there are several project management tools on the market, some targeted for more traditional project management processes such as MS Project and others that are targeted for more Agile type planning such as VersionOne, Rally, and Mingle from ThoughtWorks. The company in this paper currently uses a combination of planning tools including MS Project Server, SharePoint, and Mingle to manage their projects.

MS Project is used as a high level planning tool to gather high level estimates to produce a cost analysis for the project, for time tracking, and higher level monitoring of the project’s actuals to baseline, and to provide the project data (hours and cost) needed for the final cost analysis. Since the hourly rates of all employees are loaded in MS Project Server, in addition to tracking the actual hours being logged to tasks in a project, it provides the actual cost spent throughout the project. MS Project Server also comes with SharePoint which provides a useful platform for managing project documentation. While it provides the PMO high level project tracking, budget, and project documentation management, it has not been effective for development teams trying to manage their stories, backlogs, and planning iterations; picturing progress during the release; and producing burn-down charts. Mingle is used by the teams to manage their story backlogs, more detailed information about the story including requirements clarifications, acceptance
criteria, the testing approach that will be used for the story, and tracking of the story as it moves through
the different stages of development (Backlog, Iteration Backlog, In Development, In Test, In UAT, Ready
for Prod, Complete).

For the past year, the company has been using a combination of SharePoint lists to gather the data for the
projects the teams are working on by release and Microsoft Excel VBA modules to produce their team
utilization numbers. During release planning which forecasts the teams’ work for the next (3) months, the
development managers provide their estimates at the feature level for some projects while other projects
they just provide the percentage of their team that will be allocated to the project for that release. Although
the development managers prefer the team percentage estimate as it is a quick rough estimate, it doesn’t
provide insight into which set of features are being worked on in a particular release nor when the
particular feature is targeted to be completed. While the goal is to move the teams away from team
percentage estimate for release planning, the Release and Team Capacity Planning application can provide
a forecasted burn-down chart based on the team size, the percentage of team being allocated each release,
and the number of hours in the release. The current SharePoint/VBA solution is limited since it cannot
enforce relational integrity and is dependent on refreshing data in the spreadsheet, running a VBA program,
and then copying the data back into SharePoint lists. This solution presents opportunities for copy/pastes
mistakes and doesn’t produce project burn-down or budget burn-up charts. The Release and Team Capacity
Planning application is built on an Oracle database enforcing relational integrity and is a web application
that can be made available for use by the development managers and project managers to immediately see
charts and reports based on the release planning data input for the various projects.

Program Requirements

The high level requirements of the program include:

• Allow user to enter a team percentage estimate for a project or feature based estimate in hours for
release planning
• Enter project tracking information e.g. actual hours
• Project burn-down chart (effort)
• Project burn-up chart (budget)
• Team Capacity chart and reports

The Release and Team Capacity Planning application is a web-based create, read, update, and delete
(CRUD) application that provides a series of screens to collect, analyze, and report project, team and
release planning data. Information collected includes descriptive information about the release, the size of
the development teams per release, the effort required for a project’s iteration, and the actual hours worked
each iteration. It also allows features to be assigned to a team and release. The system provides the flexibility to allow the team to enter an hours-based estimate at the feature level for a particular release or a team-percentage-based estimate for the project for a particular release.

The team-percentage-based estimate is a requirement unique to the company and has been an area that the PMO has pushed back against because the original rough estimates provided by the teams are in hours and if they begin providing their release plan in a team allocation % format, it is easy for them to lose sight of how that breaks down in hours and compares to their original estimate. Also, since the team is just providing the percentage their team is allocated to the project for a given release, the current system does not have the information needed to determine when a particular feature in the project is scheduled to be completed. One of the benefits of the Release and Team Capacity Planning application developed in this project is that it provides charts that highlight the project trends based on the data regardless if the team is providing an hours-based estimate at the feature level or team-percentage-based estimate at the project level. The data is then used to produce a project hours burn-down chart, a project budget burn-up chart, a team capacity chart/report showing the team’s allocation across several projects, and a roadmap report of the final targeted release for the features. Appendix C contains the high level use case diagram for the Release and Team Utilization Planning application.

**Implementation**

The Release and Team Capacity application was designed and developed using all cloud based tools including Lucidchart for the use case diagram and the data model and a hosted version of Oracle Application Express 4.2 (APEX) for the development of the database and application. APEX is a declarative browser-based rapid application development (RAD) tool for developing Web-based and mobile applications using SQL and PL/SQL as the core languages for development. Using the APEX tool for the UI and report development, it generates the necessary web client side technologies including HTML, JavaScript, and CSS and jQuery Mobile to create applications for mobile devices. It also provides Oracle Application Express SQL Workshop which provides utilities to load and unload data from an Oracle database, generate DDL, view object reports, and restore dropped database objects.

Over the past five years of my career, my focus has been as a project manager of Agile projects and working with various teams to improve their iterative software development practices. Prior to that, I had three years of experience doing web development using J2EE technologies including JSP, Struts, EJB, Spring, and Hibernate along with HTML, JavaScript, CSS, and AJAX for client-side development. While becoming fairly proficient with the various technologies, at times I found the integration of the various technologies to be frustrating and felt a lot of unproductive time was spent resolving integration issues.
rather than focusing on the functionality of the application itself and delivering business value. For example, I often found myself looking for better JSP and JavaScript tools to speed up web page design, development, and debugging. Prior to the period of doing J2EE development, I spent (8) years doing Oracle database development using Oracle PL/SQL development that provided all of the business logic and database access for a Java applet and appreciated the simplicity of the framework and the speed at which we produced fully functioning applications for various clients. That said, once the framework was established, although the set up and configuration was complex and often frustrating, I appreciated the advantages of the MVC framework separating the data, business logic, and presentation layers with a more vendor independent and loosely coupled solution. Given this experience, I decided that I would need to find a tool that would allow me to develop a user-friendly, feature rich web application in a short timeframe without those integration problems noted above.

While I had never worked with Oracle’s APEX RAD tool, it interested me for this project because it appeared to be an integrated environment that would meet those needs. I was interested in focusing on producing on an application where my focus could be spent on implementing the functionality of the application to solve a real world issue versus spending that time getting the various components of a technology stack to work together. I was also interested in learning the strengths and weaknesses of Oracle’s RAD tool and leverage my knowledge of Oracle database development using SQL and PL/SQL. While evaluating Oracle APEX to determine if it looked like it would be a viable option, an early discovery was Oracle offered a free trial hosted APEX environment so I was able to avoid a lot of the time that would have been required to set up an Oracle instance and the APEX environment and was able to begin creating the application immediately.

Since APEX was classified as a RAD tool, my assumption was it was used primarily for smaller development type projects e.g. prototyping, but learned it had been used to develop large, complex, multi-lingual applications such as the Oracle Store, Oracle Cloud, and was also used to create a mobile application for America’s racing team in 2013 America’s Cup.

The APEX engine provides the following tasks (Riaz Ahmed 2013):

- rendering
- processing application pages
- session state management
- authentication services
- authorization services
- page flow control
- validation processing
- rendering and page processing
The Release and Capacity Planning application provides the data collection screens and reports that are listed and described in Appendix A.

**Results, Evaluation, and Reflection**

Oracle APEX proved to be an effective tool for gathering, analyzing, and reporting data in a web-based multiuser CRUD application. Although it isn’t necessarily a good fit for many applications or organizations, it is a useful tool for an Oracle shop with experienced SQL and PL/SQL developers to quickly build Oracle-based web and mobile applications.

The goal of the project was to provide visibility into the medium- and long-term planning efforts at the company. One of the biggest benefits the application provides is a visualization of the project trends in terms of the project completion through the use of a burn-down chart. Figure 4 shows a more effective burn-down chart produced by the Release and Team Capacity Planning application based on estimates entered at the feature level. The Ideal Remaining line provides a view of the plan if the burn-down rate was a steady rate based on the total planned hours divided by the number of iterations. The Planned Remaining line shows the projected trend based on the schedule and the red line shows the trend based on the actual hours reported. If the red line trend is above the ideal remaining, the project is behind schedule, if under it is ahead of schedule. The important thing is how the actuals are trending and can help determine the project completion based on the current velocity which is the amount of work the team is completing work each iteration, the remaining work, and the remaining time. In the example below, the project is ahead of schedule. From a budget perspective, similar to the Hours Burn-down Chart, the Budget Burn-up chart in Figure 5 provides a visual of the project’s budget trend. In this example, the project is spending faster than the ideal rate and faster than originally scheduled, but work is also being done ahead of schedule so there is no concern for now. There would be concern if the project budget was being spent ahead of schedule and the Hours Burn-down showed behind schedule.
Figure 4 - Example of More Effective Burn-down Chart (feature based estimates)
The PMO has found this application to be an effective tool. After demonstrating the application developed as part of this project to the management of the organization described in this report, the tool has increasingly become a useful tool to evaluate the trends of ongoing projects as well as in the up-front planning process. One of the immediate benefits the application provided was the insight on projects where the teams provided team-percentage-based estimates for their remaining work instead of hours-based estimates at the feature level. The PMO can now quickly and easily view the trend of the project metrics based on that data. For example, in the burn-down chart in Figure 5, we see the original estimates were for 330 hours, but based on the team’s percentage based estimates during release planning once implementation began, the total estimate grew to 383 hours, the trend of the actual hours is also above the estimate, and the projected completion date is one release past the original planned final release. After reviewing the visual burn-down chart, corrective action was taken to work with the teams to determine if the projected remaining hours was correct and if additional funding was going to be needed to complete a final requested feature. It also highlights the shortcomings of the team-percentage-based method of planning by the teams since there is no projection of feature completion and long term, the company’s plan is do feature level estimating for all of its projects.

Figure 5 - Budget Burn-up Chart
Figure 6 - Project Burn-down for EPO966
In Figure 7, the budget is projected to be $5K over based on the team percentage based planning. Again, the visual charts provided feedback that the percentage based planning estimates were higher than the original hours based estimates.

In addition to the project completion and budget trends, once the various project data has been loaded for the teams, the capacity of the teams is calculated and displayed in a chart. See Figure 8 below.
Conclusions and Future Work

While the application is useful in its current form, there are a number of enhancements that would make it more useful by eliminating some of the manual effort involved in setting up the project ideal burn-down and burn-up as well as pulling the actual hours data from MS Project data housed in a SQL Server database.

Also, adding the ability to prioritize the features across projects would assist in the release planning process and could potentially be used to produce a recommended schedule based on team capacity and prioritized projects/features. Currently the features entered are free-form text so it would be difficult to do analysis at the feature level due to inconsistencies on how each team may have entered the same feature with slightly different names.

Conclusions:
• Oracle’s APEX development tool proved to be an effective tool for developing a web based application for collecting, analyzing, and reporting data using SQL and PL/SQL.

• The graphical representations of the project burn-down and budget burn-up charts provided a valuable visualization of the project’s trends based on the velocity and remaining time and work. This representation of the data is useful for the PMO, the teams, and management and facilitates a more focused planning discussion based on the data.

• While estimates provided by the teams during release planning as a percentage of their team is not the preferred method and ineffective in terms of forecasting when particular features are completed, the Release and Capacity Planning application can at least provide more insight into the project trends and a projected project completion date when using that estimation method. This method may be acceptable for shorter-term projects (e.g. 2-3 months) where a roadmap isn’t as critical assuming all of the features defined for the project will be completed in the 2-3 months, the long-term vision is to end this practice and estimate and prioritize at the feature level.

• While this application has filled a current void for the company in this study, for the long term, the company should still investigate full featured Agile Planning commercial tools like VersionOne that can support all of the Scaled Agile Framework levels. There may be cases where the Release and Capacity Planning tool can be used where there are specific gaps because of unique practices that may not be supported by the commercial tool.

In summary, the project met its goal of providing visibility into project- and portfolio-level planning and allowed the author to gain experience using a new, efficient RAD tool for CRUD application development.
Bibliography

Appendices

Appendix A

Release and Team Utilization Planning Screens/Reports

The data collections screens include:

- **Project Information**
  - Project Name: Title of the project
  - Actual Hours Flag: Designates for team capacity planning purposes whether the team will provide hours at the feature level to calculate the team’s percentage on that project for a given release or whether the team will just provide an estimate at a team % level using the Team Utilization screen.
    - If Actual Hours Flag = ‘Y’, the program will calculate percentage of the team working on the project for the release by totaling all of the feature estimated hours for a given release and dividing by (team size * hours of a full time equivalent (FTE) for the release).
    - If Actual Hours Flag = ‘N’, the user enters the team percentage for the project for a given release. This is most often used when a team is doing analysis when the team is being allocated to work on a project, but the features haven’t been fully defined e.g. doing analysis to provide estimates.

- **Releases**
  - Release Date: Date the features will be deployed into production
  - FTE Hours: Hours a full-time equivalent (FTE) is available to work on a given release.
    As a standard, this company uses 6.5 project hours available * number of working days in the release e.g. in a 4 week release, the FTE Hours would be 6.5 * 20 = 130. 6.5 hours was chosen as the number of hours a resource would be available for project work assuming 1.5 hours/day to address administrative type tasks e.g. meetings, e-mails, vacations, etc.

- **Teams**
  - Team Name: Title of the team

- **Team Sizes**
  - Team: A specific team.
  - Release: A specific release
  - Team Size: The size of the team for that particular release

- **Release Plan**
• Team: A specific team
• Release: A specific release
• Project: A specific project
• Feature Number: A specific feature
• Feature Description: Short description of the feature
• Planned Hours: Number of hours the team is planning for that feature for that particular release.

• Project Ideal/Actuals
  • Project: A specific project
  • Release: A specific release
  • Ideal Remaining Hours: Number of hours ideally remaining for the project after this particular release. Calculated based dividing the total hours by the number of releases and subtracting that number from the total remaining each release until 0 hours remaining in the final release.
  • Actual Hours: Number of actual hours across all teams spent on the release
  • Ideal $ Spent To Date: Calculated based dividing the total budget by the number of releases and adding that number to the total budget spent each release until the total budget is reached in the final release.

• Team Utilization
  • Team: A specific team
  • Release: A specific release
  • Project: A specific project
  • Team Utilization Percent: The percent of the team to be allocated to the project in this particular release. It may be calculated based on feature hour estimates or entered at the team level depending on what method the project is using.
  • Hours Based: ‘Y’ indicates whether the percent shown is calculated based on feature hour estimates, ‘N’ indicates it was entered as a team based percentage.

Report and charts using the release planning and data utilization data are as follows:

• Team Utilization Report (shows the team capacity for all teams, projects and releases)
  • Team: A specific team
  • Release: A specific release
  • Project: A specific project
  • Percent: The percent of the team to be allocated to the project in this particular release. It may be calculated based on feature hours estimates or entered at the team level depending on what method the project is using.
  • Hours Based: ‘Y’ indicates whether the percent shown is calculated based on feature hour estimates, ‘N’ indicates it was entered as a team based percentage.

• Project Burn-down Chart
  • Release
- Ideal hours Remaining
- Actual Release Hours
- Actual Hours Remaining

- Project Budget Burn-up chart
  - Actual Release Cost
  - Ideal Total Budget Spent
  - Planned Budget Spent
  - Total Budget Spent

- Project Roadmap
  - Show milestones and final targeted release for each feature
Appendix B

Release and Team Utilization ERD
Appendix C

Use Case Diagram