



Technical Team

Members'

Roles and Responsibilities

Version 1.1 Final

Prepared by Technical Team Management

Northern Arizona University

December 19, 2014

Revision History	2
1. Introduction	1
1.1 Scope	1
1.2 Purpose	5
2. Overview	7
2.1 Team Culture	7
2.2 Resource Management	14
2.2.1 Use of Communication Tools	15
2.2.2 Schedules and Records of Absence	18
2.2.3 Sick Time	20
2.2.4 Vacations	22
2.3 Process, Procedures, and Practices	24
3. Workflow	27
4. Task Management Environment	35
4.1 TME Users	38
4.2 Projects	43
4.2.1 Internal Projects	46
4.2.2 External Projects	48
4.3 Milestones	51
4.4 Tasks	54
4.4.1 Task Attributes	55
4.4.2 Task Management and Sprint	93
4.5 Creating and Queueing Tasks	103
5. Programming Group Roles and Responsibilities	108
5.1 Resource Management	112
5.1.1 Use of Automated Tools	114
5.1.2 Schedules and Records of Absence	116
5.1.3 Sick Time	118
5.1.4 Vacations	120
5.2 Process, Procedures, and Practices	122
5.2.1 Development Process	124
5.2.2 Maintenance Process	162
5.3 Mentoring	205
6. Business Analysis Group	211
6.1 Resource Management	213
6.1.1 Schedules and Records of Absence	222
6.1.2 Sick Time	224
6.1.3 Vacations	225
6.2 Design Processes	227
6.3 Video Support Processes	235
6.4 Project, Milestone, and Task Management	237
7. System Architecture and Software Engineers group Roles and Responsibilities	311
7.1 SASE Management	315
7.2 Database Management	316
7.3 Server Systems Management	320
7.4 Software Architecture Management	324
7.5 Internal Project Management	331
7.6 Resource Management	333

Revision History

Name	Date	Reason For Changes	Version
Dennis Spurlin	20140917	Initial Draft	Draft 2.0
Dennis Spurlin	20141031	Added Sections 6 and 7	Draft 2.1
Dave Dennehey	20141117	Modified Section 5	Draft 2.1
Amanda Kapp	20141126	Modified Section 7	Draft 2.1
Dennis Spurlin	20141201	Modified Section 4	Draft 2.1
Dennis Spurlin	20141205	Added Section 2.2	Draft 2.2
Kyle Cawood, Jason Robinson, and Dana Stoneberger	20141215	Completed section 6	Draft 2.3
Dave Dennehey, David Brink, Kevin Hayes	20141216	Edited section 7	Draft 2.4
John Meyer, Marc Colvin, Dennis Spurlin	20131219	Completed section 7, fixed typos, and repaginated.	Final 1.1

1. Introduction

1.1 Scope

The Extended Campuses Technical Team (ECTT) under Marc Lord, Technical Director, is composed of four groups. The System Architects and Software Engineers (SASE) group, supervised by Kevin Hayes, Applications Systems Analyst Programmer Lead, is responsible for system administration, database administration, and system architecture. The Business Analysis group, supervised by Kyle Cawood, Business Analyst Lead, is responsible for project management, graphic design and video production. The Programming group, supervised by Dennis Spurlin, Senior Applications Systems Analyst Programmer, is responsible for software maintenance, applications development, and systems applications supporting version control, integration, migration, and deployment of production capabilities. The Computer Support group, supervised by Lisa Young, Coordinator Senior, is responsible for desktop hardware, software, and network connectivity for all of Extended Campuses statewide. This document describes the roles and responsibilities for the first three groups only. It does not address the roles and responsibilities of the members of the Computer Support group.

1.2 Purpose

The purpose of this document is to describe the roles and responsibilities of Technical Team members, with respect to the organizational structure established within Extended Campuses (EC) for management of new and existing Web application capabilities maintenance and development. Detailed descriptions of the team's established processes and procedures are specified in this document. Specific definitions for projects, milestones, tasks, and other concepts are needed to clarify their use within the team's established processes and procedures. This document is a living document, modified as needed within our rapidly advancing technology.

2. Overview

2.1 Team Culture

The Engineering, Business Analysis, and Programming groups work closely together as a single team with each member individually performing his or her role without the confusion of overlapping responsibilities. Constantly changing technology requires a team effort to develop specialized expertise in different areas. Each member is unique with regard to his or her knowledge and skills. Therefore individual research into options for possible technical solutions for new development is a necessary activity, and unconstrained collaboration is essential to the success of the team. This adds to an ongoing workload with occasional time constraints that require a balance between structure and flexibility.

Structure is necessary for a team to function. However, it must be tailored to the organization's mission. It is not possible to support systems that must function twenty-four hours a day for seven days a week with a team available only between the hours of 8 AM and 5 PM. Therefore, flexible schedules are part of the team's culture. The focus is on getting the work done without processes and procedures that unnecessarily get in the way. For example, some industry standard practices have been ignored with regard to restricting access to modify production capabilities. It must be noted that industry standard practices are proven effective ways to solve problems. Therefore, some of these practices have been incorporated into the team's concept of operations.

Our team has the distinction of pursuing agility and accountability in our development and maintenance activities, either by design or by accident. In some cases, we learned that our "out of the box" thinking was already a proven practice, and in other cases we realized the limitations of thinking outside the box. Most of the problems we will encounter already have industry standard solutions. Three sources provide an abundance of methodologies suited to our pursuit of solutions. "Agile Software Development Ecosystems" by Jim Highsmith (2002), "Kanban, Successful Evolutionary Changes for Your Technology Business" by David J. Anderson (2010), and "Agile Project Management" by Jim Highsmith (2013) are must reads for team members. A balance between structure and flexibility considers that some standard practices would be helpful but require more resources than is available. Others may not be applicable to our goals. Only those that can be beneficial and easily adapted to our flexible environment should be considered. Our flexibility has proven to be a most valuable characteristic. However, management of the work requires tracking and accountability in order to report progress on a regular basis. The Task Management Environment (TME), developed by the team, provides ways to manage accountability and tracking of the workflow. The TME, along with other processes and practices contribute to the roles and responsibilities of the programmers, engineers, and business analysts. But before describing the TME, let's look at our workflow.

2.2 Resource Management

Collaboration is key to the efficient flow of work in an IT environment. The resources available to us for communication are the phone, email, instant messaging, office space, conference rooms, and the TME. The need for effective communication increases as the impact of our products becomes more critical to the success of Extended Campuses and the team's growth and development can only be achieved by our universal participation in speaking and listening to each other.

2.2.1 Use of Communication Tools

Each team member is provided with the resources needed to perform their functions. Some of the applications pre-installed on each personal computer are required by NAU for data security. Others are required for communication. In this day and age, professional success will depend on compliance and competence with the use of the technologies provided by the employer. Team members are responsible for ensuring that approved applications needed to perform their functions are installed, working properly, and used in accordance with the team members roles and responsibilities whether working at their desk or at a remote location.

The Technical Team uses Lync and Outlook primarily to manage our communication and the TME to manage progress on projects and tasks. Team members must be available whether in the office or out. Either way, access for needed communication must be specified in a way that the entire team can know when and how to collaborate.

2.2.2 Schedules and Records of Absence

Weekly schedules for part time team members are posted and maintained in SharePoint by their supervisors. Part time team members also submit an online Louie bi-weekly timesheet each pay period to be approved by their supervisor. Weekly schedules for fulltime team members are posted and maintained by each member in Outlook. This can be accomplished by first going to Outlook→File→Options→Calendar→Work Hours. By setting the earliest start time and the latest end time for the weekdays, the Outlook calendar will appear unshaded for those work hours. Exceptions for specific days of the week can be limited by creating an appointment in the unshaded hours not scheduled to work. All time off, for vacation or when sick, is recorded using the online PeopleSoft application for Record of Absence (ROA) for fulltime members. For complete guidance on time off reporting, see “[Payroll Information](#).”

2.2.3 Sick Time

Key team members (supervisor and project managers) are to be notified prior to missing scheduled work hours when sick, if possible. Sick time may be made up, but required attendance at meetings cannot and require a Record of Absence (ROA) for fulltime team members. Emails to key team members are expected to keep them informed about resource availability. ROAs for sick time are due upon return to work.

2.2.4 Vacations

Requests for days off by fulltime team members are submitted as Records of Absence (ROAs). Plans for time off should be entered as ROAs two months in advance whether plans are tentative or firm. Prior to the departure of a fulltime member, making them unavailable for more than two days, they shall coordinate with all stakeholders,(ASAs, BAs, designers, developers, etc) on all projects with tasks in progress.

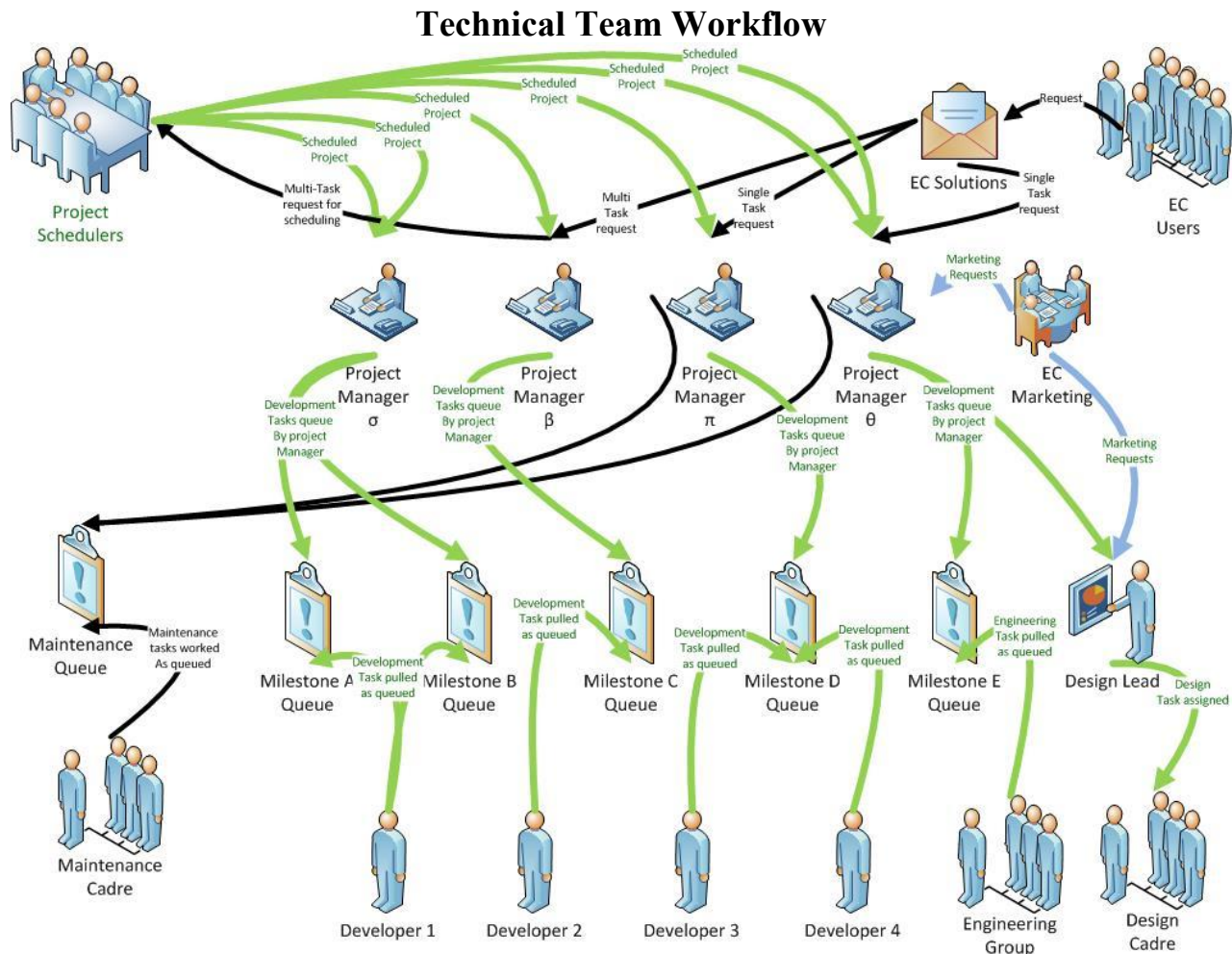
2.3 Process, Procedures, and Practices

The Technical Team uses formal processes to perform routine development and maintenance functions. These processes are automated whenever there is an obvious advantage to doing so. Occasionally, procedures are modified to accommodate the automation and improvement of our processes. Every effort should be made to update this document and others to reflect new procedures and processes. This is also true of the software development practices used by the programmers and the project management processes used by the business analyst and engineers. See section 5.2 for a description of the programming processes, section 6 for a description of the business analysis and project management processes and section 7 for a description of the engineering processing. Issues related to processes and procedures should also be formally recorded and reported. This can be done through weekly reports. All fulltime team members are required

submit a weekly 515 report to their supervisor using a format and on a schedule specified by their supervisor.

3. Workflow

Work for the technical team actually has four sources. The most visible source is from upper management. The project schedulers decide what new capabilities they want for EC and generate projects for the business analysts to analyze in conjunction with the developers. These new capabilities and major enhancements to existing capabilities, requiring multiple tasks, fall into the category of “Development.” The second source comes from maintenance- if current production capabilities or resources which we use malfunction and require immediate resolution. A third source of work is from EC Marketing. Their requests may go to a business analyst or directly to the design lead. The fourth source of work is for internal maintenance and administrative activities. This work does not go through the project schedulers, but directly to the project managers in the category of “Maintenance.” Internal maintenance work is managed by the engineers. Administrative actions are managed by supervisors and individual team members.



The previous diagram shows the routing of external requests. Maintenance requests from EC users are depicted using black lines for the flow to the business analysts, and then assigned directly to the

maintenance chief for the maintenance cadre of programmers to pull from. A separate task flow for production issues allows development work to be grouped into iterations that can be integrated independent of the maintenance migration path to production. The EC Marketing team requests to a business analyst or to the design lead are depicted with blue lines.

This diagram depicts the task flow process for development work with green lines. When the project schedulers decide to proceed on a project that includes software development, the project is given to a specific technical team business analyst to manage the software development aspects of the project. Business analysts coordinate with developers to determine which developer(s) will work on which project milestones. Business analysts also work with developers to create and sequence tasks for each of their milestones so that they may each be viewed in a queue by milestones. The queued tasks are ordered by severity with the lowest numbered severity to be worked first for each project. Developers pull tasks from the top of the queues for milestones they are working on. Once tasks are placed in the status of “In-Testing” they no longer count as one of their active tasks. They may pull a second task from the top of the queue to have in reserve to work when needed. When the project manager of a task changes the status to “On-Hold”, it also does not count as an active task. Once a task reaches seven days past assignment without being worked, it is considered to be “languishing,” if it is in progress, assigned, on hold, or in testing. Tasks should never be allowed to languish.

4. Task Management Environment

As you can see by the workflow diagram, the process would be a challenge to manage without the ability to track and report on the flow. The Task Management Environment (TME) is one of the tools needed to do that. It does a lot more than just track tasks. It provides assistance to business analysts and engineers to manage projects. Other processes used are the sprint, weekly TME meetings, scrum meetings, and weekly 515 reports. All work performed by the team can best be managed as projects. Projects may be organized into milestones as described below. Analysis can be aided by using the milestone features of the TME and tasks can be created with the requirements and objectives documented at the project level, milestone level, and the task level. Tasks are all related to a project and may also be related to a milestone for a project. The TME provides facilities for planning projects, organizing them into milestones, and creating, tracking, and documenting progress on tasks. Documents may be associated with projects, milestones and tasks to provide a researchable store of information related to the work we do. The TME is a core component of our task management reports and our sprint process. Different team members use the TME in different ways. This section describes the TME and the features used for creating and queueing tasks.

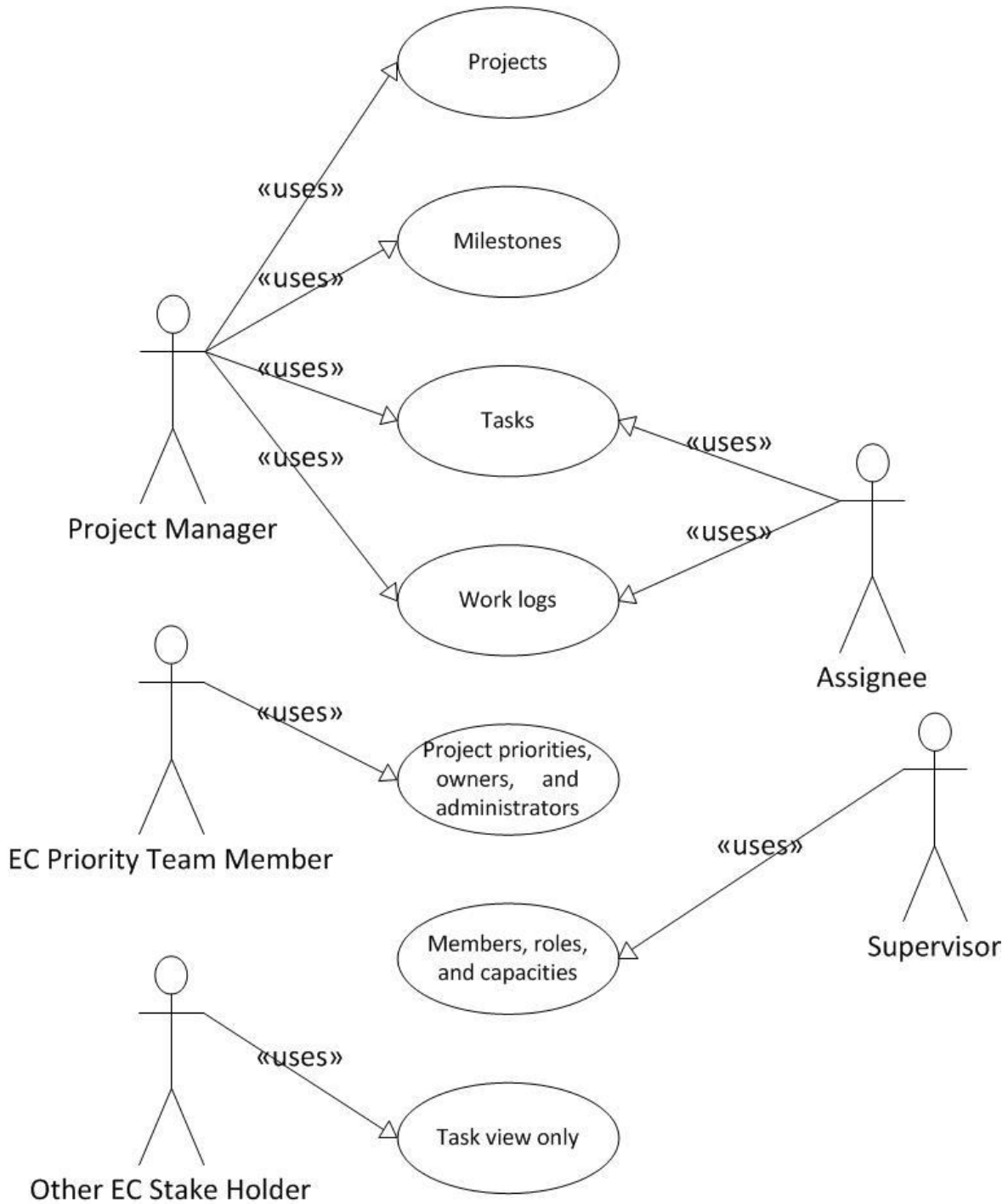
4.1 TME Users

The following use case diagram provides a description of the TME’s seven functions and the relationship with the five users.

Project managers use projects, milestones, tasks, and work logs. They manage projects and create tasks.

Assignees use tasks and work logs to log work progress and update tasks.

Supervisors add and remove team members and set roles and capacities.



EC Priority/scheduling team members set priorities, owners, and administrators.

Other EC stake holders may only view active tasks.

Project managers have a major role in the TME. They convert requirements for development and maintenance into tasks. Regardless of the project or the source, only the project manager for the systems addressed in the request initiates action and performs whatever coordination is needed in order to create one or more projects, milestones, or tasks.

4.2 Projects

Projects generally support two different areas of EC. Projects that support only the technical team are internal projects. Most tasks for internal projects have a cause of “Internal Administration.” Projects that support specific departments, other than the technical team, are considered external projects. External projects for development of new capabilities, applications, or Web sites are referred to as “development” projects. External projects for maintenance of current production capabilities and internal automated processes are referred to as “maintenance” projects. These distinctions guide the creation of projects and management of tasks created, to follow distinct processes. Projects for development of new capabilities generate tasks with a cause of “New”, but are automatically changed to “Sprint” at the start of the next sprint. Projects for ongoing maintenance of existing capabilities generate tasks with a cause of “Maintenance.”

Projects shall have a title and a description that specifies the scope and purpose of the project. Projects are always associated with a specific project manager. They may also be associated with a specific department, a folder location for storage of project documentation, and other EC staff members as owner and administrator. Project managers create projects by providing a title, description, and documentation of the requirements and scope of the project. It is highly recommended that project managers attach all documents related to the project. Assignments of owners, administrators, and priorities are done by the EC priority team, if needed.

4.2.1 Internal Projects

Internal projects support the technical team, and as such, EC in general. They have no specific external customer. Nor do they need EC staff members as owners and administrators. Therefore, they are never assigned a priority. For example, a server maintenance project would be an internal project because it supports EC in general and the technical team in particular. Another example would be a performance review project for tasking supervisors to do annual reports. They have no need for an owner, administrator, or priority because the project manager has sole responsibility for the project. All tasks for these internal support projects are in support of the technical team department. Internal support tasks do not normally need to follow a sprint process unless it has specific upper management interests. In those cases, the cause would be “New.”

4.2.2 External Projects

External projects support EC management, supervisory, and operational staff outside of the technical team. These are our external customers. If project support is provided to customers external to EC, there should be an EC member acting as liaison to function as the EC customer. These external customers may coordinate with the EC upper management project priority team for priority, owners, and administrators as needed. Development projects generate tasks that are for new capabilities (systems not already in production) and have a cause of “New.” They are tracked

in the sprint process as development. Maintenance projects generate tasks for current production capabilities. If a task for maintenance can be completed as a single task migrated to production it has a cause of “Maintenance” and is tracked in the sprint as maintenance. Otherwise it has a cause of “New” and is tracked in the sprint as development.

4.3 Milestones

Milestones are always associated with a specific project and inherit attributes from that project. The attributes are the priority as project priority (not severity), the department, and the administrator. All three of these attributes may be null. Milestones require a title when created. They should also be given a description and have all related documents, specific to the milestone, attached to a folder location referenced to the milestone. The optional required-by date is only needed when all tasks for the milestone absolutely need to be done by a specific date.

New development projects may be analyzed into multiple milestones that define sequential phases of development or concurrent threads of development. The purpose of a milestone is to assist in the definition and requirements of an enhancement or new capability to be used in a group of tasks that can be created from the milestone. Milestones are only used by project managers and can be useful in many ways. The use of the milestone function is highly recommended anytime it is expected that multiple tasks need to be completed and tested as a group prior to deployment to the production environment. Milestones may also be used for categorizing maintenance of production issues.

4.4 Tasks

Tasks may or may not be associated with a milestone, but they are always associated with a project. Tasks are created for projects by their project managers. Project managers select options to include a project and a department, or they can create it from a milestone that is already associated with a project and perhaps a department. Most tasks will require multiple work log entries.

4.4.1 Task Attributes

Tasks have many attributes allowing the creator of a task to specify the work to be done for the task, as well as how the task will be processed. The following attributes are used to define tasks so they are tracked to an efficient completion. Whenever possible attributes are set by default, but can be subsequently changed. However, some attributes are more tightly controlled and other attributes are set by automated actions during the processing of a task.

4.4.1.1 Assignee

Upon creation, the default assignee is the user creating the task. When tasks are queued by their creator, they become available to be pulled according to their severity. Tasks are pulled by a team member by reassigning it to themselves. The list of possible assignees includes all members of the technical team.

4.4.1.2 Attachments

The option for a link to a network folder location on the file server is provided for any emails, query scripts, configuration files, diagrams, and documents related to tasks, milestones, and projects.

Documents are primarily uploaded by project managers, but assignees often find valuable documents that need to be added to the task attachments folder. Attachments provide easy access to documents related to tasks, milestones, and projects for reference by all users of the TME.

4.4.1.3 Cause

Cause is a required field that task creators must select. The options are “New”, “Sprint”, “Maintenance”, and “Internal Administration.”

4.4.1.4 Creator

The default for this attribute is the user creating a task and it is not editable thereafter. Creators can be any member of the technical team.

4.4.1.5 Department

EC departments are listed in the DLS database, Admin_department table. Projects may or may not be associated with a specific department. However, milestones and tasks shall always be associated with an EC department. If not null, the department ID in the project table is used to populate the milestone and task tables by default, but may be manually selectable. Tasks created from milestones are populated with the milestone’s department ID.

Projects are not always in support of a specific department. When a project can be associated with a department, the related milestones and tasks shall inherit (duplicate) the association. However, occasionally tasks are associated with a different department than is inherited from a project or milestone. So milestones and tasks shall allow selection of a department when none is in the project or when the department is different from the project or milestone.

4.4.1.6 Description

Description for projects, milestones and tasks are optional, but critical to efficient completion of work. Descriptions are distinctly related to the project, milestone, or task and specify the requirements and scope of each. Tasks created from milestones inherit milestone descriptions and require further description specific to task requirements. Additional details may be documented in the attachments for projects, milestones, and tasks.

4.4.1.7 Due Date

Task due dates are automatically set at two weeks from the current date when created, and again when the status is changed to “Assigned.” They may not be manually edited unless they have a cause of Internal Administration.

4.4.1.8 Entry Date & Time

The default for this field is the date of creation and cannot be subsequently edited.

4.4.1.9 Estimated Cost

This is a required field at task creation, defaulted to 8 hours, editable at the time of creation and not editable thereafter. Traditionally, the unit used for estimating task cost has been hours. A precision of four hours is sufficient. Estimates in 4 hour increments should be approached from a perspective of assurance that the task will take less than the estimate. Sprint tasks are to be limited in size so as not to overload the assignee. Therefore, the recommended options for estimating a cost of a task

with a cause of “New” and “Sprint” are 4, 8, 12, 16, and 20. The preservation of the initial estimate provides an essential metric for the improvement of estimates.

4.4.1.10 Milestone

Tasks may or may not be associated with a milestone. Milestones offer project managers tools for analyses and decomposition of projects and major enhancements. Creating a maintenance task for a project without using a milestone option is an efficient way to get a simple issue solved. If the task is one that can be ready for deployment upon completion and not dependent on other tasks for its deployment, then a simple task is a good approach. However, when developing enhancements or phases of a project, tasks cannot be deployed until all dependent tasks have been developed and integrated. The best way to manage groups of related tasks to be deployed simultaneously is to use milestones.

4.4.1.11 Phase

The task phase attribute is a list of categories that can be used to catalog tasks into groups that can be measured separately. The purpose is to gain metrics to improve task estimation methodologies. Studies of these metrics, over time, are expected to yield insight into a better list of attributes. Ongoing refinement and study of this task phase attribute depends on improved task specification and analysis, to decompose the categories more distinctly.

4.4.1.12 Project

All tasks are required to be associated with a project. Project managers create milestones for their projects and tasks for their milestones and their projects. A task created from a milestones is automatically associated with the project for the milestone.

4.4.1.13 Project Manager

This field is located in the project table. Each project is associated with only one manager.

4.4.1.14 Project Manager’s Note

This attribute is only accessible for edit by the project manager. It is a short note associated with the task to be used at the project manager’s discretion.

4.4.1.15 Project administrator

This optional field may be located in the project table. Each project is associated with no more than one administrator if any. This is determined only by the EC Priority Team.

4.4.1.16 Required By Date

This is an optional field. This is only needed when there is a “Drop Dead” date on a task or milestone. It is not the due date. In other words, if the task is not done by this date, it will have been futile. Tasks shall inherit a required dates from a milestone’s “required by” dates when available so that all tasks created from a milestone have the same “required by” date.

4.4.1.17 Severity

This is different than the project’s priority. Priority carries an assessment by management with regard to development projects. EC management assigns numeric values to projects to establish their importance to management. Severity is the technical team’s way of specifying which tasks are to be completed first. Severity of tasks for development projects use a range of integers that depend

on the length of the queues. The tasks to be done first are assigned the lowest severity numbers for each project queue.

Tasks for maintenance work only require a range of integers from 1 to 5 as an assessment of the operational impact of a maintenance task. The following table provides an easy way to determine the severity of maintenance tasks.

Maintenance Issue Severity

Production Capabilities Work-Around	Functionality	Severity
No	Adverse	1
Yes	Adverse	1
No	None	1 or 2
No	Limited	2
Yes	None	2 or 3
Yes	Limited	3
None needed	Minor Error	3 or 4
None needed	Cosmetic Issue	4 or 5

The most critical issues are those that impact the database with incorrect data whether or not there is a work-around. Capabilities that no longer function and have no work-around cause a work stoppage. These tasks may be critical or high severity depending on the impact on the overall operation. Any issue that is either non-functional or for which there is no work-around should be at least a medium or high severity. Other issues are low or minimal severity depending on the impact of the minor issue.

4.4.1.18 Start Date

This date is programmatically set to the current date upon task creation for each task. It is automatically updated to the date the task is changed to the status of assigned. It is not manually editable. The trigger for resetting the start date is on the condition that the assignee is changed while the task is in the status of either initial review, backlog, or queued.

4.4.1.19 Status

The initial status of tasks upon creation is “Initial Review.” The normal flow for changes in status goes from “Initial Review” to “Queued” or to “Backlog” then “Assigned.” Once assigned, a log entry automatically triggers a change to “In-Progress.” Manual changes by the assignee to “In-Testing” and then to “Final Review” completes each task. Final review tasks are automatically changed to “Closed” at the end of each sprint. There are eleven options for task status:

- Initial Review: This status is for initial evaluation of prioritization, sizing, and requirements prior to assigning the task. All tasks are created with this status automatically.
- Queued: Used as task inventory by project managers for production and new development tasks.
- Backlog: Used by task creators to hold tasks in reserve for possible use later.
- Assigned: Tasks in “Assigned” status have been pulled or assigned for work to begin, but have yet to be worked on. Tasks are automatically changed to this status upon initial assignment.

- In Progress: Once work has begun on a task, it is automatically changed to this status. The first work log entry following the initial assignment shall trigger this status.
- In Testing: Assignees select this status just prior to acceptance or integration testing depending on whether the task is for production or new development.
- Final Review: Assignees select this status once they have verified that the task has been deployed to the production environment or integration testing proves successful, depending on whether the task is for production or new development.
- Closed: Project managers shall never close a production and new development task. Only the task manager is responsible for closing production or new development tasks.
- Canceled: Only the project manager for a task shall change the status to “Canceled.”
- On-Hold: Only the project manager for a task shall change the status to “On-Hold.”
- Ongoing: This status shall only be used for internal administration tasks and not for production or new development tasks.

4.4.1.20 Title

The title is a required field for projects, milestones, and tasks. Task titles may be automatically populated from milestone items, but it must be entered if not associated with a milestone item. The title provides a distinctive way to reference a project, milestone, or task without having to use its ID number. It is not a description and should fit easily on a single line.

4.4.1.21 URL

This is an optional field where a link to a Web page may be inserted. The Web page may be on one of our sites where there is an error, or it may be to a location that clarifies the task.

4.4.1.22 Work Log

Changes in the task assignee, status, track, severity, and due dates are automatically recorded in the work log. Assignees shall also record significant comments about their work and the amount of time spent on tasks.

4.4.2 Task Management and Sprint

Individual tasks are managed by the project managers. The role of the task manager is to track and report on work in progress as a whole. The task manager is concerned with the sprint process and procedures that maintain a sustainable flow of work, and providing project managers with the information they need to size and coordinate the completion of tasks with respect to their projects. The technical team uses the agile concept of a sprint in a special way. Rather than “sprinting to completion”, we use our sprint as a regular interval of time to measure progress. Since we have concurrent responsibilities for maintenance of several projects, and new development for high priority projects, a single sprint involves progress for many diverse projects. This regular interval facilitates the collection of meaningful metrics. The use of the sprint in this manner is critical to our processes regardless of what we call it. Each sprint represents the weekly progress of work assigned during the previous week. New tasks pulled each day of a current sprint are limited so as to not exceed each developer’s capacity. For this reason, it is crucial that any work performed on new and sprint tasks is logged within a day of being performed so as to gauge progress on tasks. At the beginning of each sprint, all assigned “New” tasks automatically become “Sprint” tasks and progress on projects is measured with respect to task estimates for completion dates and work load.

The sprint provides a regular interval for visualizing progress on projects, tracking individual tasks, and managing work in progress.

The management of tasks for both maintenance of existing capabilities and development of new capabilities requires separate processes. The industry standard best practice solution is to have separate organizations perform these separate functions. Often a government or corporate customer will manage the maintenance of their own capabilities on their own hardware and contract out the development of any new capabilities. The fact that we do both can be a blessing and a curse. It requires that we maintain separate processes for development and maintenance on the one hand, but allows us the choice of when we use which on the other hand.

Maintenance is defined as modifications to existing capabilities made to correct errors, address changes in business rules, and enhance the performance and usability of applications that the organizational units depend on to perform the functions. Development is defined as the creation of new capabilities that do not affect any existing capabilities that the organizational units depend on to perform their functions. The reason that different processes are needed is to avoid impacting the existing capabilities during development of new capabilities. If we consider this reason alone as the guiding factor in our decisions of what projects should follow a maintenance process and what projects should follow a development process we can take advantage of flexibility not offered in the industry standard solution.

The primary cause of impacts to production capabilities during development of new capabilities has to do with the magnitude of the effort. Most maintenance efforts are addressed by a single task that can be migrated though the test environment to the production environment without any dependence on other tasks. The automated CC.Net migration process has always supported this approach to migration. If a single task can be migrated without dependence on other tasks in the works all the way to the production environment, then the maintenance migration path is the most convenient to use regardless of the distinction between maintenance and development. However, if there are multiple interrelated tasks that must be migrated simultaneously, these tasks must be integrated in a separate environment and shown to be functional prior to being introduced to the maintenance migration path.

The decision to use the development migration path should be made early in the project planning stage by the maintenance chief when there is a likelihood that the development will impact the maintenance of existing capabilities. More information on the details of these processes is provided in section 5.

4.5 Creating and Queuing Tasks

Requests from the project scheduling team, EC staff, and team members are associated with an existing project, or a new project is created. If the request will require multiple tasks, the project manager has the option of creating one or more milestones associated with the project to assist with the analysis and creation of tasks. However the tasks are created, they are initially assigned to the project manager creating it in the status of “Initial Review” with a two week due date. The creator of the task must select the cause for the task. If the task is created as a single maintenance task, the

project manager selects a cause of “Maintenance.” Tasks created for internal use and not for an external customer, such as administrative training and process improvement, have a cause of “Internal Administration”. When creating one of many tasks needed to develop a new capability, the creator must select a cause of “New.” The cause of “Sprint” is never selected. It is set automatically by the TME to begin tracking. The project manager also has the option to select a task phase. The task phase option is a list of categories for cataloging types of tasks. This list provides metrics for process improvement.

At this point, the project manager may leave the task in their open task list in the status of “Initial Review” or change the status to “Backlog” or “Queued.” Putting tasks in “Backlog” status removes the task from the project manager’s open list and saves it for later action. Putting tasks in the status of “Queued”, adds them to the queues of tasks ready to be pulled for their related projects. Once tasks are queued, they become available to be pulled by team members selected to work on the project. Tasks are pulled by team members when they select themselves as the assignee. The status is automatically changed to “Assigned” and the due date updated to two weeks from the date assigned. Tasks remaining in backlog or queued status shall have the due dates updated at the beginning of each sprint to 14 days from the current date. Project managers work with other team members to have their tasks pulled for programming, design, and systems work on their projects.

5. Programming Group Roles and Responsibilities

In the role of supervisor of the programming group, Dennis Spurlin (ASA Programmer Sr), is responsible for conducting scheduled reviews and appraisals of performance, accounting for missed time, reporting work performed, and ensuring that members of the programming group follow NAU policies. The supervisor of the programming group also fills the role of task manager. As task manager, he monitors the flow of tasks for development and maintenance, collects and reports metrics on individual and team productivity, and coordinates collaboration for process improvement proposals.

Damien Coy (ASA Sr), performs the role of lead developer. As lead developer, he consults with the chief software engineer, Vivek Bongu, and with the BAs on software requirements related to the tasks created by the BAs for to be worked by the development cadre. He is also the point of contact for the maintenance chief, Dave Dennehey, when assistance is needed on maintenance issues.

The fulltime programmers and apprentice programmers perform work and tasks. Tasks pulled for programming work involve specific processes and practices for successful coding, versioning, integrating, testing, migrating, and deploying software solutions. The roles and responsibilities of programmers require adherence to standards and process, ongoing collaboration and training, and participation in collaborative teamwork.

5.1 Resource Management

Programmers assume responsibility for completing tasks they pull from the task queues, but the work is often done in a collaborative manner. Each programmer brings to the team specialized skills as well as skills shared in common with the team. It is important to recognize that the

differences between the programmers strengthens the team. They should collaborate with each other freely and collaborate with the project managers on the tasks they work to ensure efficient completion of quality work. Communication with available resources is key to effective teamwork. In order to facilitate an environment conducive to effective teamwork, the following tools and practices are programmer responsibilities.

5.1.1 Use of Automated Tools

Programmers will need to maintain access to communication resources to ensure their availability to the team. Programmers are also responsible for downloading and installing approved applications needed to perform their functions. In addition to Lync and Outlook to manage communication and the TME to manage progress on projects and tasks, programmers need access to SQL Management Studio, Visual Studio, and other tools not included in the standard NAU image. The TME, Outlook, Lync, and the telephone are essential to the job whether sitting at a workstation or working remotely. Programmers should start each day by: checking email and calendar events, ensuring TME work logs are current, and verifying connectivity through Lync and other means of communication. A programmer's availability for work is determined by his or her status reported in Lync even when sitting at their workstation. Other team members working remotely may be waiting to converse regardless of location.

5.1.2 Schedules and Records of Absence

As stated above in section 2.2.2, weekly schedules for fulltime programmers are posted and maintained by each programmer in Outlook. Weekly schedules for apprentice programmers are posted and maintained in SharePoint by the programming supervisor. Time off, for vacation or when sick, is recorded using the online PeopleSoft application for Record of Absence (ROA) for fulltime programmers. Part-time student wage apprentice programmers submit an online Louie bi-weekly timesheet. Section 2.2.2 also describes how to schedules are posted using Outlook for full time members. Programmers need to let team members know that they are available as scheduled by appearing available in Lync.

5.1.3 Sick Time

As stated above in section 2.2.3, team members with a stake in your work (supervisor and project managers) are to be notified prior to missing scheduled work hours when sick, if possible. Sick time may be made up, but required attendance at meetings cannot and require a Record of Absence (ROA) for fulltime programmers. Emails from fulltime and apprentice programmers to key team members are expected to keep them informed about resource availability. ROAs for sick time are due upon return to work.

5.1.4 Vacations

As stated above in section 2.2.4, requests for days off by fulltime programmers are submitted as Records of Absence (ROA) two months in advance whether plans are tentative or firm. Prior to the departure of a fulltime programmer, making them unavailable for the majority of a sprint (ending each Tuesday), they shall coordinate with the project managers on all tasks for which they are

assigned, including work being done by apprentice programmers. When apprentices are sick or on an approved absence for the majority of a sprint, the mentors of tasks being worked by apprentices assume responsibility for coordinating the work the apprentice was doing.

5.2 Process, Procedures, and Practices

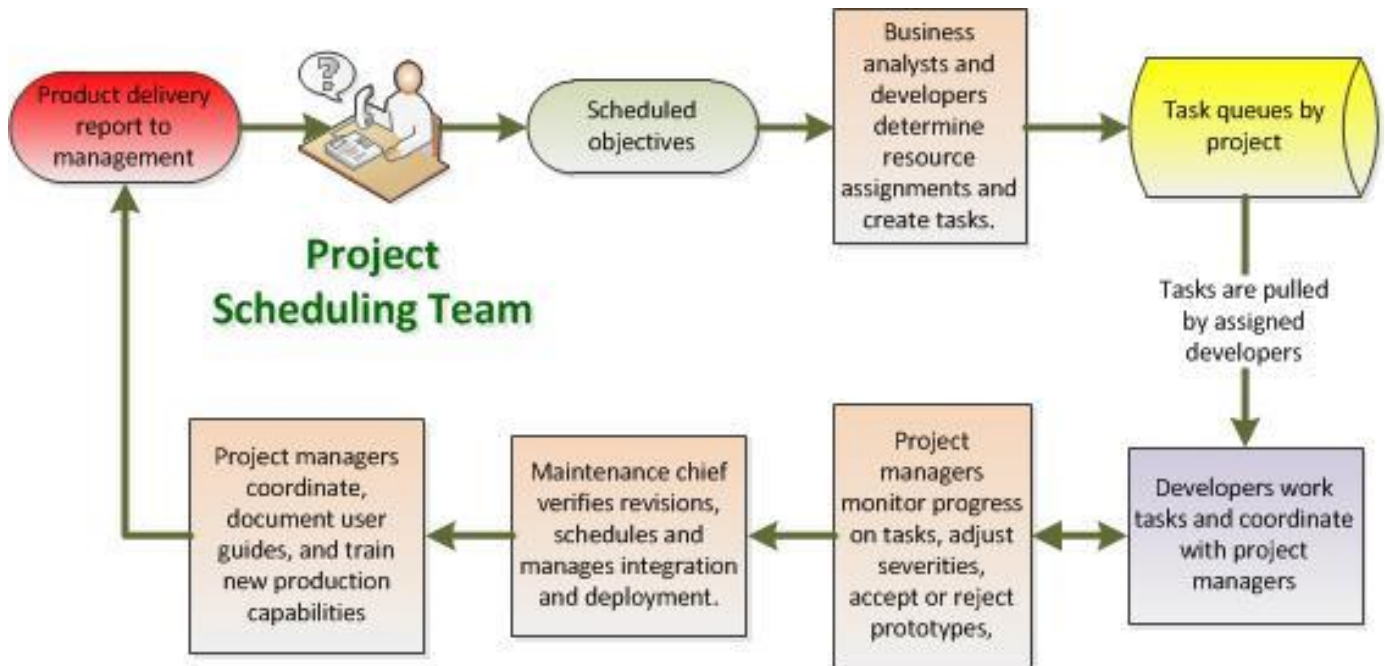
Fulltime programmers in the programming group submit a weekly 515 report to their supervisor due by close of business on Wednesday of each week. The 515 report can be generated using the TME. Task progress is monitored using the TME in concert with the two week overlapping sprints starting at the close of business on Tuesday of each week. Maintenance and sprint tasks pulled by programmers are due within two weeks of the date pulled. Once pulled, work on a task should begin within seven days or it is considered to be languishing. Each programmer is limited to no more than two sprint or maintenance tasks at any one time, so proper focus can be given to standards for the quality and quantity of work. In addition to following standards set by the engineering group for systems configurations, database schema, coding practices, and the use of third party tools, programmers follow specific and distinct procedures for migration of development and maintenance projects. These two separate processes are described below, one for development of new capabilities, and one for maintenance of existing capabilities.

5.2.1 Development Process

Development processes are distinctly different than maintenance processes. There is a specific order to developing a new capability, much like making a car. There are plans drawn, configuration and specifications determined, and the preparation of the assembly line. With maintenance of a car where the car is already built, it's put on a lift, and a part is simply changed out or adjusted. There may be some confusion when it comes to enhancements to existing capabilities. The best rule to use for enhancements is: "If it takes multiple tasks to be complete before it is ready to use, treat it as development."

Development projects are scheduled by the project scheduling team. Business analysts create and queue tasks in sequence to be pulled by a development programmer. Development programmers may occasionally integrate their development artifacts in folders separate from those used for maintenance of production capabilities in the event that it is determined we cannot persist both development and maintenance through the pre-established migration paths and servers. Once a fully functioning capability is approved by the project manager for migration, it is merged to the test branch and tested for acceptance by the project manager. If needed, the maintenance chief can assist with this operation. If accepted, it is deployed to production and if not it is immediately reverted from the test environment. The following diagram shows that development work is scheduled by the project scheduling team, managed by the business analyst, and worked by the development cadre of programmers.

Development Workflow



5.2.1.1 Development Task Processing

The TME is used for all tasks. This paragraph describes task processing by programmers for development tasks. These tasks have “New” and “Sprint” causes only. Applications Systems Analysts (ASA) and apprentice programmers never create tasks for external requests. However, they do create tasks for internal and administrative purposes. When creating an internal administrative task, please follow these tips:

1. Enter a short unique title that covers the scope of the task.
2. Enter 0 (zero) for severity.
3. Select “Internal Assistance” for cause.
4. Select the “Technical Team” department.
5. If the task is an ongoing task, select Status of “Ongoing” and enter estimate of 2000 hours.

Tasks for external customers are created by business analyst. They create tasks with a short unique title, specified department, detailed description, and hopefully enough attached documents to adequately describe what needs to be done. Some requests can be satisfied by a single task, while others require many tasks. A request for a bug fix, minor change in a business rule, or any other change to a production capability that can be done is a single task and migrated to production without dependence on any other tasks is a maintenance task which follows processes described in paragraph 5.2.2. Large requests for new capabilities or products require analysis and decomposition into functions or, in agile terms, “stories.” These new capabilities or products that require multiple tasks follow the developmental migration path.

Tasks are pulled based on each programmer’s task load and require TME log entries to be current since tasks are pulled on a daily basis. Reports are generated throughout each sprint and require updated TME entries subject to upper management review. Progress on projects is determined by differences between estimated sprint tasks durations and hours worked on tasks. Log entries should inform the project managers of accomplishments and obstacles and include any significant

information related to meetings and discussions clarifying requirements. Critical information such as the pathnames to related digital artifacts and Subversion revision numbers must be included in the TME work logs. Here are three ways to enter TME work logs.

1. From the View Task clock – This is the fastest way to make a single log entry. Any team member may enter their work to any task by clicking the clock next to the task number, selecting date, entering hours to the nearest half hour, entering significant comments, updating the status if needed, and saving. Team members do not have to have the task assigned to them just to enter a work log.
2. From the Quick Task page – Use this approach if additional documents need to be attached to the tasks. You may create a folder by clicking the attachments button. If the folder does not open, you can copy and paste the network location from the field next to the button into “Run” command. Comments and hours worked can then be logged by clicking the clock below the Update button for Work Logged, select date, enter hours to the nearest half hour, enter description of work performed and issues dealt with during the time period, update the status if needed, and save.
3. From the Work Log page – This option is only available for tasks that are actually assigned to the user. It is useful when making multiple entries for multiple tasks to save time. Select the task at the top of the page before beginning to log work to each task worked. Select date, enter hours to the nearest half hour, enter description of work performed and issue dealt with during the time period, update the status if needed, and add. Repeat as needed.

When creating tasks, the project managers are responsible for the Title, Department, Requester, Severity, Cause, Project, Task Phase, Project Leader, Estimated Duration, and initial Description of each task for an external project. Programmers in the development cadre may coordinate the use of resources in the maintenance cadre with the maintenance chief. When an apprentice programmer is assigned work by a programmer from the development cadre, the fulltime programmer becomes a mentor with respect to the task (See paragraph 5.3, on mentoring). Fulltime programmers for all tasks for external projects are responsible for the following:

Attachments: Assignees and apprentice programmers working on tasks should copy all emails and documents related to a task to the attachments folder of the task. This includes emailed statements of task approval, SQL queries sent to the DBA for deployment, and clarifying documents like diagrams, images, or shortcuts to other related folders.

Description: Assignees may make minor additions/changes to the description as needed with project manager coordination.

Status: Tasks are in “Assigned” status until the assignee begins to log work. Tasks are placed “In Progress” once work begins. Once the assignee has unit tested a task and merged to special integration folders in the development branch of the repository, they notify the project manager that the task has been completed and then they put it in “Final Review” with the project manager’s concurrence. Special tasks are required for migrating groups of

integrated tasks to the test environment by the maintenance cadre. Development tasks are never migrated to the test and production environments by members of the development cadre.

Work Logs: All actions related to task progress are entered in the task work log focusing on accomplishments and issues using the procedures described in the options for work log entries above. These entries are important for work load determination and for smooth transition from state to state. For example, contacts with project managers for approval of integrated tasks should be noted in the logs. The assignee must also log all of the relevant Subversion revision numbers they have committed to the development branch in a work log for that task. Likewise the maintenance chief needs to log all relevant revision numbers from the Subversion development branch to the test branch and trunk to the TME.

5.2.1.2 Development Migration and Deployment

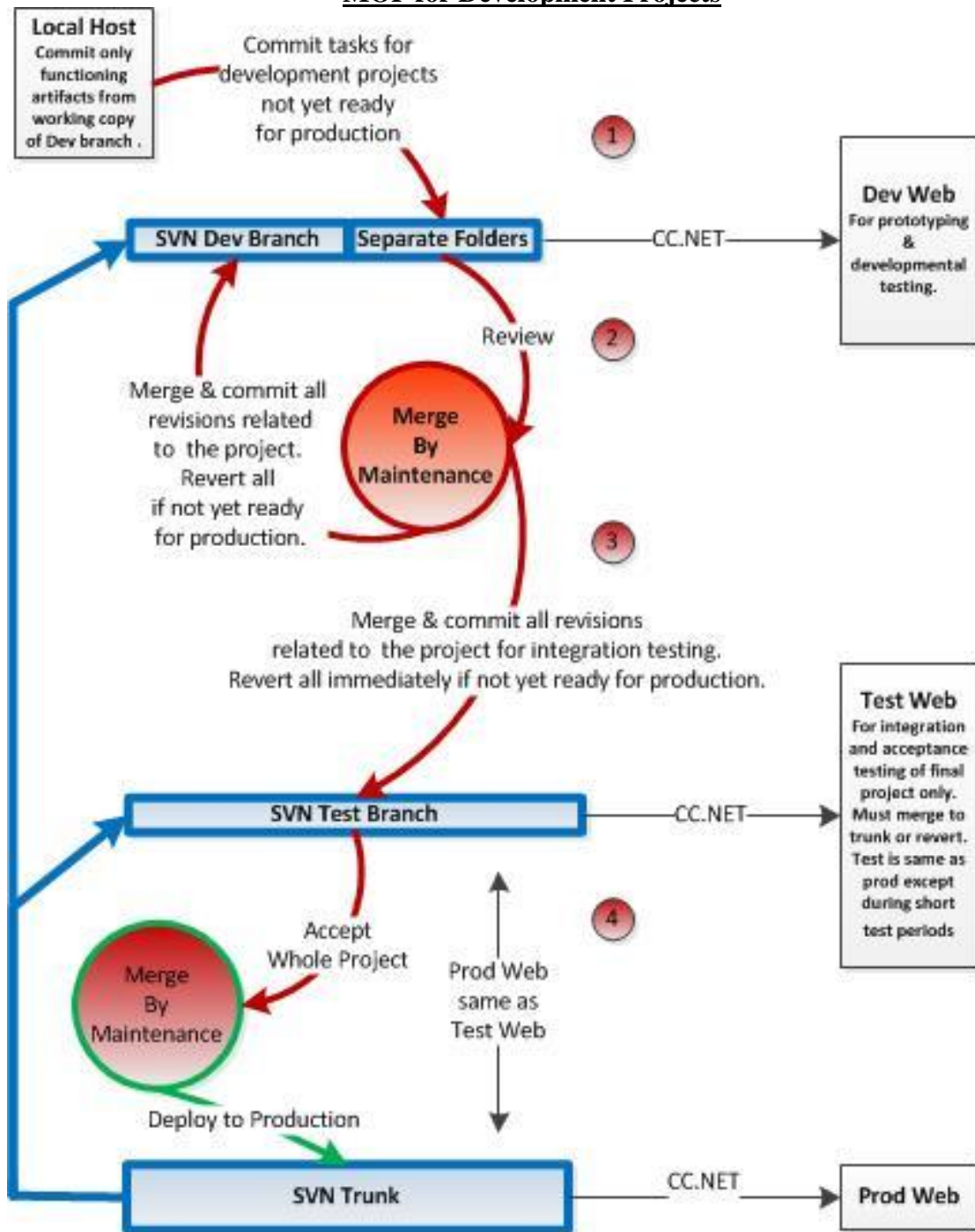
Applications maintenance and software development processes are distinctly different. Maintenance involves modifications to an existing capability to correct, update, or adjust for a minor change in business rules that can typically be done in a single task by one developer. Development is the creation of a new capability or to replace an existing capability to satisfy new requirements, often requiring several tasks and multiple developers.

The Migration Operational Process (MOP) is a tiered integration and migration process comprised of three permanent environments. The process involves an automated integration application triggered by changes to a version controlled repository with the help of command files to assist with the migration of changes through the tiered environments. The development, test and production environments use Cruise Control (CC.NET) to build and deploy directly from the Subversion repository to each respective Web environment. Programming tasks usually involve artifacts modified in Visual Studio and unit tested locally by the assignee. After the assignee is reasonably sure that requirements are met, the artifacts are committed to the Subversion development branch for automated migration to the development environment.

The following diagram represents how the MOP is used for development projects.

1. Programmers initially develop on their localhost and commit to the development environment only artifacts for completed tasks that have been successfully tested on localhost. This provides easy access for project managers to engage in prototyping and verification of results before proceeding to formal acceptance testing. It also ensures that their work is consistent with the work of other programmers. The development repository and Web site environment may require preparation prior to beginning development of a project. Existing production capabilities rely on the integrity of the development branch repository and Web site for ongoing maintenance. Separate folders in the development repository and separate IIS environments must be used when needed to protect the integrity of development environment for maintenance to also continue without obstruction. CC.Net should additionally be configured to deploy to a separate development Web environment for prototyping and developmental testing when needed. Coordination of building these resources is coordinated via the maintenance chief.

MOP for Development Projects



- The maintenance chief may coordinate with other fulltime programmers for reviewing code, verifying revision numbers, coordinating migration with the maintenance cadre, and participating in integration testing with the project managers as needed. When a functional iteration is ready for production, the migration begins with the development branch.
- Some phases of development require that partial integration testing be done in the test environment. When needed, the test should be conducted immediately after committing to

the test environment and then immediately reverted from the test environment after completion to maintain consistency between the test environment and production. When ready for production, migration begins with the development branch all revisions for the milestone or project are committed to the test branch for acceptance testing. If the test results are rejected the revision is reverted from test and development branches. Steps should be taken to avoid obstructing maintenance operations. Upon acceptance, merging to production should be immediate in order to maintain consistency with the test environment.

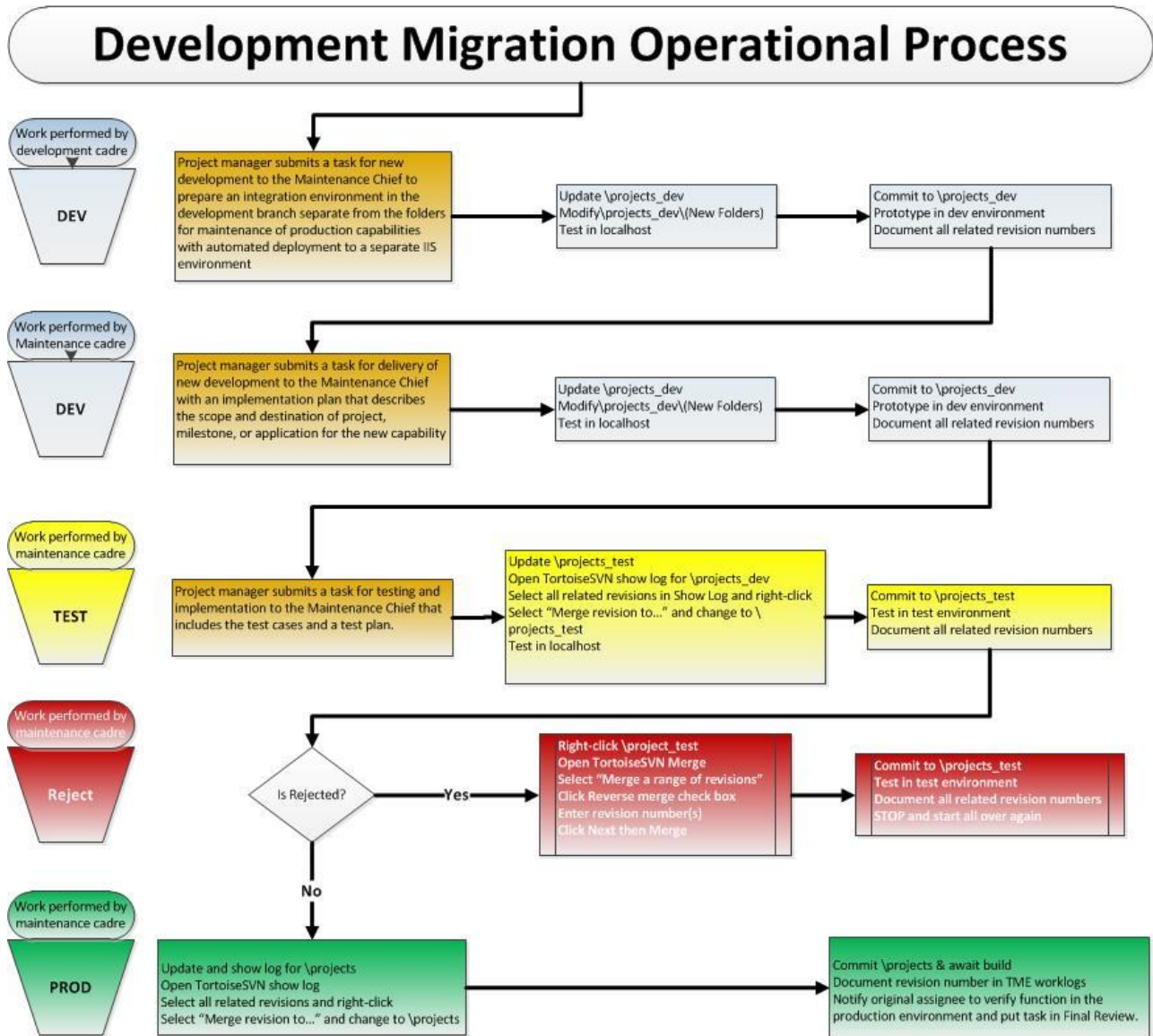
4. Deployment to production is managed by the lead developer and verified by the maintenance cadre/project managers.

Extra care needs to be taken with the migration of development projects and major enhancements so as to avoid impacting existing production capabilities. The main factor affecting the risks is whether or not the development is for an existing capability or a totally new capability not already in production. Maintenance of current production capabilities supports the day-to-day operation of EC and in some cases, NAU as a whole. The development and test environments that support those production components must also remain available on a day-to-day basis. Development of new capabilities cannot be allowed to impact the ongoing maintenance needed for day-to-day operations. Routine changes to existing capabilities are for maintenance projects. The primary characteristic of a maintenance task is that it can be migrated from the development environment to the test environment and then deployed without any dependence on other tasks. Any effort to make a change to an existing capability that requires multiple tasks to be tested prior to deployment constitutes a development project. This includes the creation of new capabilities/Web sites and major enhancements.

Tasks for development of new capabilities may require additional folders and Web configurations to avoid conflicting with maintenance processes. The maintenance chief oversees the preparations needed to begin development on new capabilities. This is best initiated with a task from the project manager to the maintenance chief to prepare an integration environment with automated deployment to a separate Web environment so as not to impact the maintenance processes.

The bulk of the work for developing new capabilities is performed by the programmers of the development cadre with whatever help they can get from the maintenance cadre. Their tasks can be put in the status of “Final Review” once the project manager has approved the results in the integration environment. Once the project manager has approved a fully functioning capability as ready for production, another task to the lead developer to work with maintenance chief if needed, will initiate the test and production deployment processes.

The following flowchart represents key decisions and steps involved with the migration of development projects and milestones.



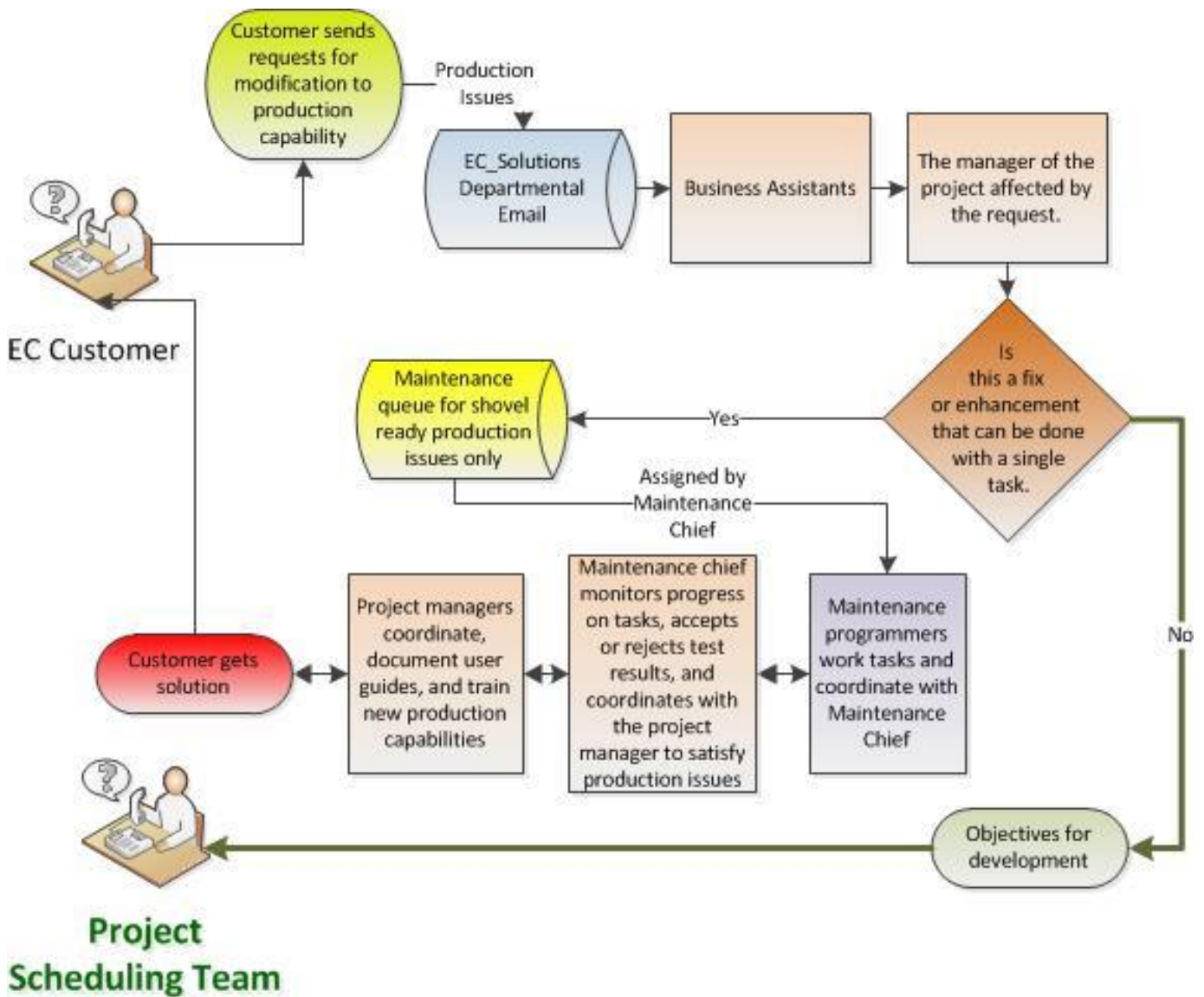
New products may also require configuration changes to associated servers and program interfaces that are not consistent with the current production environment. These inconsistencies need to be documented and solutions for replication or reversion prepared prior to migration to the test environment and to the production environment. A documented implementation plan that includes configuration changes as well as the Web application artifacts is essential to successful deployment of a development project.

5.2.2 Maintenance Process

Requests for maintenance generally come from our EC customers as emails sent to EC Solutions. Customer requests for modifications to the production environment are evaluated by the business analyst managing the maintenance project. If the scope of the request is beyond simple maintenance, such that it requires interdependent tasks to be developed and integrated prior to successful migration and deployment, then the request is forwarded to the project scheduling team.

Otherwise, the project manager creates a task and assigns it directly to the Maintenance Chief immediately. Maintenance of current production capabilities supports day-to-day operations. Maintenance tasks are defined as taking software, services, applications, etc from a broken state to a working state. General upkeep also tends to fall into this category- often applications haven't had a full management interface created for the users, and changes must be made manually. Examples of this include new data to be added to production (where not managed by the DBA), text changes (where not manageable by Ektron), and uploading files or other small, one-off, requests which are *not* feature additions.

Maintenance Workflow



5.2.2.1 *Maintenance Task Processing*

The TME is used for all tasks. This paragraph describes task processing by programmers for tasks with a cause of “Maintenance” only.

Tasks for external customers are created by business analyst. They create tasks with a short unique title, specified department, detailed description, and hopefully enough attached documents to adequately describe what needs to be done. Requests for major enhancements to capabilities currently in production requiring multiple tasks, need to be developed and integrated using a separate environment within the development environment before migrating to the test environment. Therefore, these requests should follow the development process previously described.

Maintenance tasks are assigned to the maintenance chief according to the severity provided by the project manager (See table on page 41 for details on assigning severity). The maintenance chief selects members of the maintenance cadre to the work tasks and adds them as assignees to tasks. Reports are generated throughout each sprint and require updated TME entries subject to management review. Progress on maintenance tasks is monitored on a weekly basis in the sprint. Log entries should inform the project managers of accomplishments and obstacles and include any significant information related to meetings and discussions clarifying requirements. Critical information such as the pathnames to related digital artifacts and Subversion revision numbers must be included in the TME work logs. Here are three ways to enter TME work logs.

1. From the View Task clock – This is the fastest way to make a single log entry. Any team member may enter their work to any task by clicking the clock next to the task number, selecting date, entering hours to the nearest half hour, entering significant comments, updating the status if needed, and saving. Team members do not have to have the task assigned to them just to enter a work log.
2. From the Quick Task page – Use this approach if additional documents need to be attached to the tasks. You may create a folder by clicking the attachments button. If the folder does not open, you can copy and paste the network location from the field next to the button into “Run” command. Comments and hours worked can then be logged by clicking the clock below the Update button for Work Logged, select date, enter hours to the nearest half hour, enter description of work performed and issues dealt with during the time period, update the status if needed, and save.
3. From the Work Log page – This option is only available for tasks that are actually assigned to the user. It is useful when making multiple entries for multiple tasks to save time. Select the task at the top of the page before beginning to log work to each task worked. Select date, enter hours to the nearest half hour, enter description of work performed and issue dealt with during the time period, update the status if needed, and click “Add.” Repeat as needed.

When creating tasks, the project managers are responsible for the Title, Department, Requester, Severity, Cause, Project, Task Phase, Project Leader, Estimated Duration, and initial Description of each task for an external project. Fulltime programmers for all tasks for external projects are responsible for the following:

Attachments: Assignees copy all emails and documents related to a task to the attachments folder of the task. This includes emailed statements of task approval, SQL queries sent to the DBA for deployment, and clarifying documents like diagrams, images, or shortcuts to other related folders.

Description: Assignees may make minor additions/changes to the description as needed with project manager coordination.

Status: Tasks are in “Assigned” status until the assignee begins to log work. Tasks are placed “In Progress” once work begins. The assignee unit tests their tasks on their localhost and only merges to the maintenance folders in the development branch of the repository when assumed to be ready for testing. After verifying full functionality of the task in the development environment and coordinating with the maintenance chief, the task is put into the status of “In-Testing.” The maintenance chief coordinates acceptance testing and deployment of all maintenance tasks before changing the status to “Final Review.”

Work Logs: All actions related to task progress are entered in the task work log focusing on accomplishments and issues using the procedures described in the options for work log entries above. These entries are important for work load determination and for smooth transition from state to state. For example, entries should explain changes to “On Hold” or “Backlog” status. Also, contacts with project managers for clarification of bug issues or enhancement requirements should be noted in the logs. The assignee must also log all of the relevant Subversion revision numbers they have committed to the development branch in a work log for that task. Likewise, the maintenance chief needs to log all relevant revision numbers from the Subversion development branch to the test branch and trunk to the TME. Apprentice programmers are responsible for updating work logs for task they work on as well as adding their revision numbers from Dev to the task log.

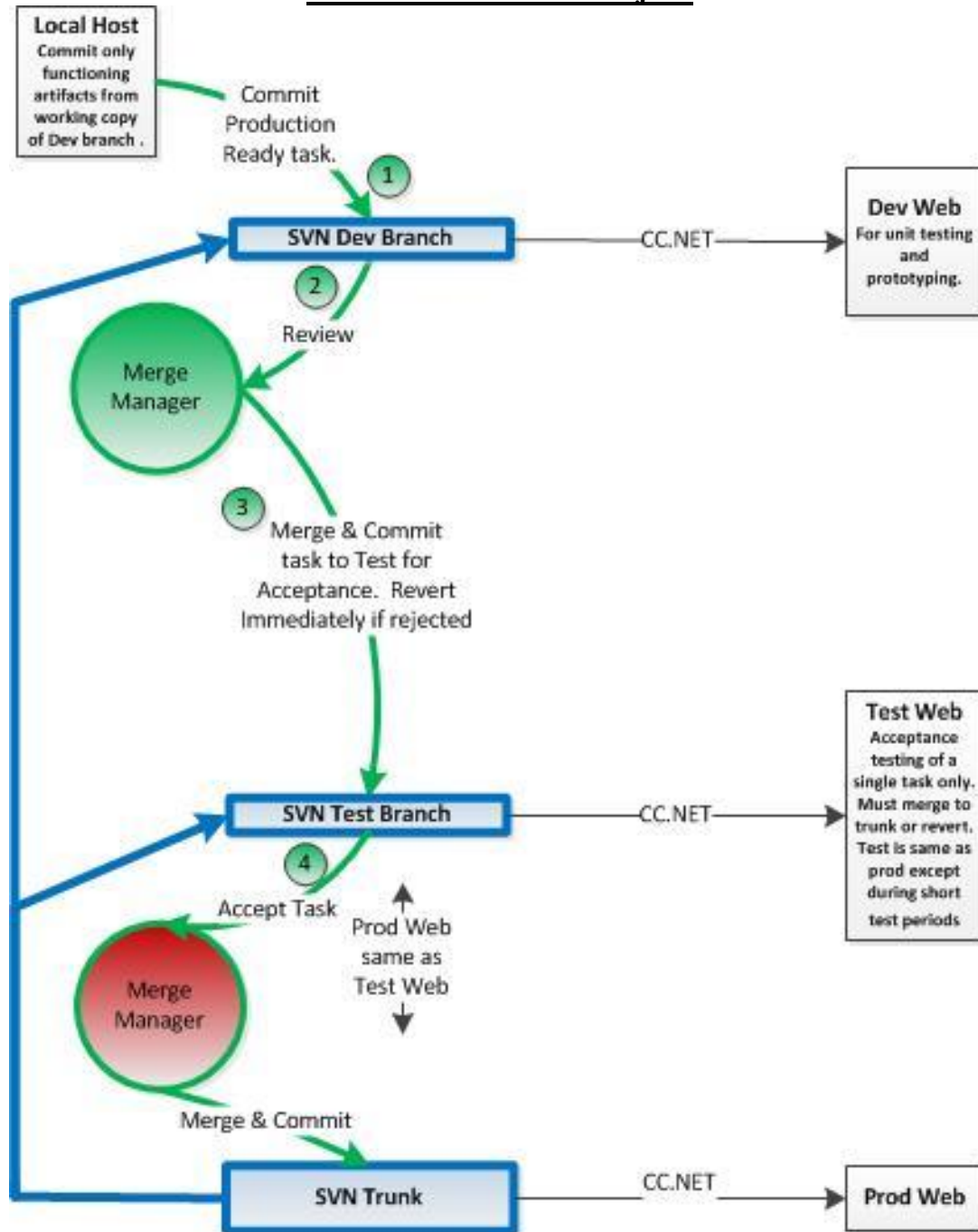
5.2.2.2 Maintenance Migration and Deployment

Since, maintenance involves modifications to an existing capability to correct, update, or adjust it for a minor change in business rules, these requests can typically be done in a single task by one developer. These modifications also follow a slightly different path through the MOP. The MOP is a tiered integration and migration process comprised of three permanent environments. The process involves an automated application triggered by changes to a version controlled repository with the help of command files to assist with the migration of changes through the tiered environments. The development, test and production environments use Cruise Control (CC.NET) to build and deploy directly from the Subversion repository to each respective Web environment. Programming tasks usually involve artifacts modified in Visual Studio and unit tested locally by the assignee. After the assignee is reasonably sure that requirements are met, the artifacts are committed to the Subversion development branch for automated migration to the development environment.

Routine changes to existing capabilities can be migrated from the development environment to the test environment and then deployed without any dependence on other tasks. Any effort to make a

change to an existing capability that requires multiple tasks to be tested prior to deployment as a group must follow the development process.

MOP for Maintenance Projects



The previous diagram represents how the MOP is used for maintenance projects. The circled numbers are described below.

1. Programmers initially develop on their localhost and commit to the development environment only artifacts for completed tasks that have been successfully tested on their localhost. This provides easy access for project managers to engage in prototyping and

verification of results before proceeding to formal acceptance testing. It also ensures that their work is consistent with the work of other programmers. The development repository has folders that correspond with the folders on the production Web site. These folders are exclusively used for maintenance. Existing production capabilities rely on the integrity of these development branch folders for ongoing maintenance. Maintenance programmers commit their changes to the development branch after they fully demonstrate the functionality specified in the task on their localhost. Verifying the functionality in the development environment ensures that their changes are compatible with other changes in progress.

2. The maintenance chief, with assistance from other programmers, is responsible for code review, merging, and verifying the revision numbers.
3. The maintenance chief also schedules testing with the project manager and migrates changes to the test environment for acceptance testing. When ready, the test should be conducted immediately after committing to the test environment and then immediately reverted from the test environment or deployed to the production environment after completion to maintain consistency between the test environment and production.
4. Deployment to production is managed by the maintenance chief with the assistance of selected development programmers as needed.

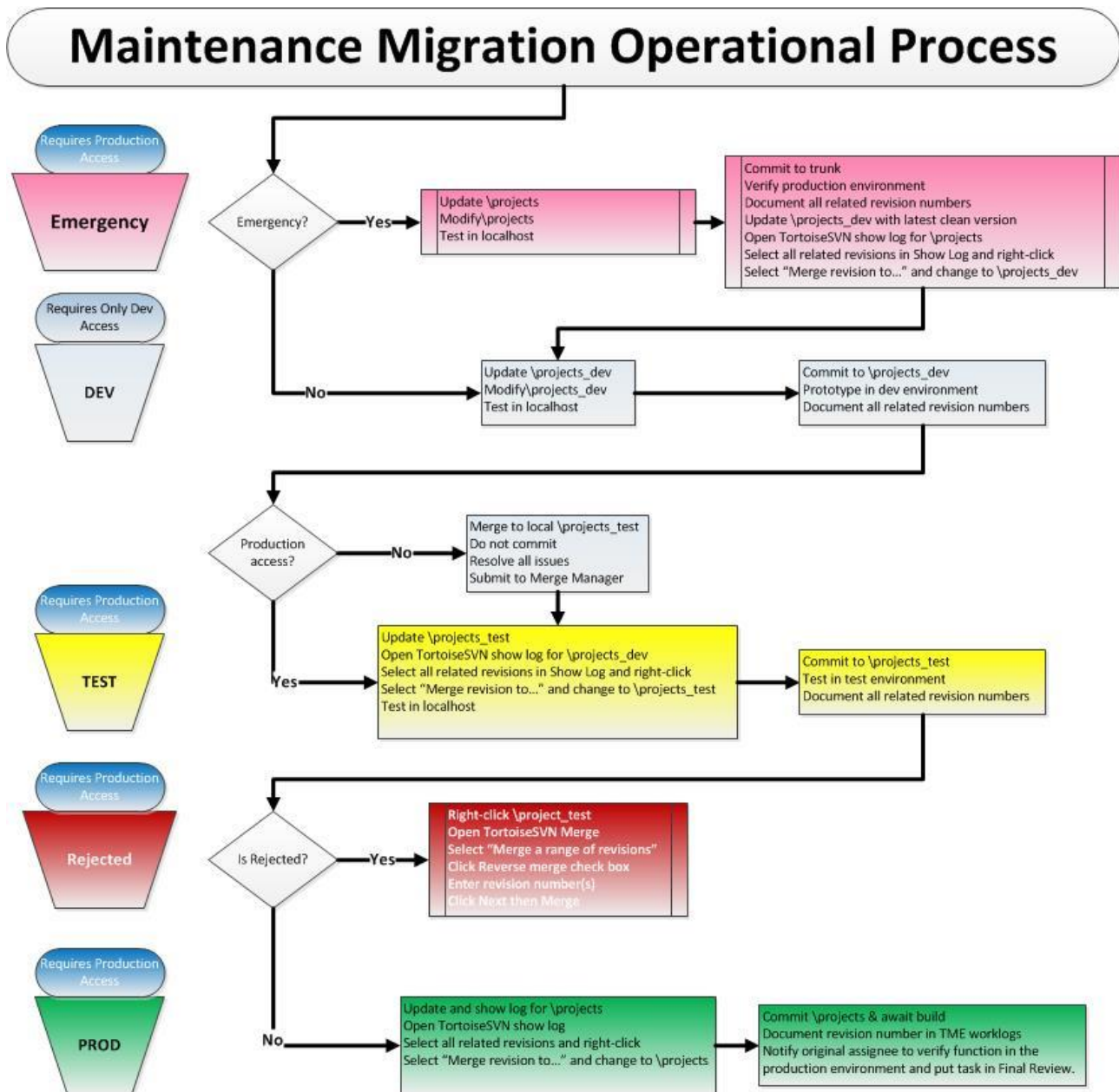
Critical maintenance tasks, if declared an emergency, may be committed directly to the trunk. However, this creates an immediate instability that requires a complete migration through all three tiers. A full migration must be completed for a second deployment to production to restore consistency within the environment tiers. The result is that emergency deployments cause additional work in the long run and should be avoided if possible.

Normally, the migration of maintenance tasks is pretty straight forward. The process involves three steps, one from localhost to the development branch, on from the development branch to the test branch, and finally from the test branch to the trunk. There are conditions to be met and other steps to be taken with each migration.

Development Branch: All of the functionality called for in a task should be demonstrable in localhost before committing the artifacts for the task to the development branch. Once committed, the assignee can work with the project manager to do preliminary prototyping and checking in the development Web environment. Any subsequent commits for the task should be documented in the task log so that all revision numbers for the task are recorded. When prototyping is completed, the assignee notifies the maintenance chief that revisions are ready for merging to the test branch.

Test Branch: Before merging task revisions from the development branch to the test branch, acceptance testing is coordinated and scheduled with the project manager for the task. Just prior to the scheduled acceptance test, the maintenance chief merges task revisions from the development branch to the test branch and monitors the acceptance testing. If the test fails, the maintenance chief restores the test environment by doing a reverse merge. If the test is successful, the maintenance chief prepares to merge the revision from the test branch to the trunk.

Trunk: The trunk of the repository deploys to the production environment. The maintenance chief merges revisions that have been successfully tested from the test branch to the trunk immediately after acceptance by the project managers.



The previous flowchart represents key steps and decisions involved with the migration of maintenance tasks. Maintenance includes rare situations where a correction is needed for operational capabilities that must be completed as soon as possible. The result of an emergency implementation is an increase in work for the maintenance programmers because after implementation to the production environment, the implementation is repeated following the process through the three tiered environments.

The first phase of the process is to consider an emergency implementation. Hopefully the answer is no and the process can move on to phase two of the process. If not, the emergency implementation is managed by the maintenance chief or designated representative.

Emergency implementations begin by developing solutions on the maintenance programmer's localhost using the development database. When confident (or hopeful) that the solution is ready for deployment, the maintenance chief checks it into the trunk for automated deployment. This carries obvious risks and should be avoided if at all possible.

Regardless whether or not an emergency implementation was needed, the DEV phase, colored gray above, is always used for all maintenance tasks. The maintenance programmer modifies their own working copy of the \projects_dev folders and unit tests their modifications on their localhosts. If all goes well, they commit their changes to from their working copy to the Subversion repository, update the TME logs and notify the maintenance chief that the solution is in the development environment.

The maintenance chief takes it from this point and uses the following process. It should be noted that only fulltime programmers have access to migrate to the test and production environments. The maintenance chief is responsible for all deployment to the test and production environments, but may delegate other fulltime programmers on a case by case basis to perform the processes. In those cases, the designated representative of the maintenance chief must also be instructed on requirements for reviewing the code and verifying revisions while acting on behalf of the maintenance chief.

5.2.2.3 Internal Maintenance

Maintenance of internal automated processes used to facilitate and manage customer development and maintenance requests require projects managed by members System Architecture Support Engineers (SASE) team. SASE project managers follow processes in accordance with those for maintenance and development as appropriate. In other words, if a change to technical team internal processes can be done by a single independent task, it follows the maintenance process. Otherwise, it follows the development process. Resources available to SASE project managers are limited to members of the maintenance cadre unless coordinated with the Business Analyst Lead.

5.3 Mentoring

The Application Systems Analysts are at the core of the team's software development productivity. In addition to performing as developers, engaged in coding, they participate in the establishment of software architectural and design standards, they evaluate adherence to standards of quality, and suggest training when appropriate. They research the latest technologies and develop training. They serve as merge managers, coordinate acceptance testing, and deploy to the production environment. They also function as "mentors" for students working as apprentice programmers. Apprentice programmers help maintain EC Web sites using Microsoft ASP.Net, C# and SQL, and third party Web development tools.

This mentoring structure allows for focused technical training opportunities for student workers and improves the management skills and increases the capacity of the fulltime staff. It encourages communication between team members about coding practices, standards, processes, and procedures.

Mentors' retain all responsibility for task work performed by apprentices. They track revisions, merge revisions for testing or integration, and coordinate with project managers. They review work performed by apprentices and provide training to ensure compliance to standards and practices. Mentors also conduct training, research new technologies, and explore the possible use of development tools and methods that keep the products provided by the team technically current in addition to development and maintenance of our Web sites and internal maintenance of our automated development processes.

Only fulltime programmers may pull programming tasks with causes of "New" and "Maintenance" and only tasks causes of "New", "Sprint", and "Maintenance" are reported in the manager's report. Fulltime programmers may provide task numbers for apprentices to log their work on tasks, but the fulltime programmer mentoring the task remains the assignee responsible for the task. No tasks with causes of "New", "Sprint", or "Maintenance" are to be assigned directly to apprentice programmers. Task with a cause of "Internal Administration" and without an external customer may be assigned directly to an apprentice programmer.

6. Business Analysis Group

The Business Analysis (BA) group lead by Kyle Cawood, Business Analyst Lead, is responsible for all initial interactions with the EC customers. While projects are managed by members of both the SASE group and the BA group, the SASE group only manages projects internal to the technical team. The BA group manages projects for external customers, performs design work, and supports the EC streaming video processes for NAU TV and other NAU agencies. The designers support EC Marketing as layout designers and all of EC as Web designers under the guidance of Jason Robinson, Senior Web Developer. There are also part-time business assistants that perform transcription and other support for our streaming video service under the guidance of Dana Stoneberger, Business Analyst Senior.

6.1 Resource Management

BAs are typically assigned to projects and work with those projects across the stages including: development, maintenance and enhancement. This allows BAs to have a historical understanding of a project and continuity in communication with the projects owners. However, most, if not all, of our projects have implications for other projects. BAs are encouraged to reach out to one another when development in one area may affect technical or business processes of another project. The BA group meets weekly to discuss progress on projects, troubleshoot obstacles and walk through deliverables.

Three of the BAs tend to work more closely on development projects within EC. One BA focuses more of her time on reporting across EC projects. As the Business Intelligence project has evolved it is apparent how each new development should be considered from a reporting perspective. Again, the BA group meets weekly and can have these conversations on a per project basis.

EC staff from around the state send requests for new work and report issues to the EC.Solutions email account. The business assistants forward these on to the BA Lead who forwards them on to the BA responsible for the related project for follow-up if the BA is available or on to another BA if the primary BA is not.. Again, this is where shared knowledge across the projects is useful.

New development is prioritized by the Scheduling Team. BAs are typically assigned new development projects on a two-month schedule. If research is required before a project can be developed, the BAs are usually assigned a period of time to scope the functional needs of the project with the project owners and scope the technical needs of the project with the Technical Team. Projects that require designs are also allotted time for the designers to create designs and communicate with both the BAs and the project owners as the design evolves.

Once the project is ready for development the BAs typically work with a member of the Technical Team to create a list of tasks and estimates for those tasks. The BAs organize projects, milestones and tasks in the TME. The BAs match various development projects with programmers for the two-month development period. The programmers pull tasks from the milestone queues and log their work to each task. Additional tasks can be made if necessary per the programmers request. BAs monitor the progress of tasks and milestones over the course of the two-month development period modifying priorities as needed depending on progress, obstacles and deliverables.

Each BA and Designer summarizes the progress made on each project as well as a summary of all the meetings they attended in a weekly 5-15 and submits this to the Business Analyst Lead. BAs and Designers are encouraged to point out any roadblocks preventing progress in the 5-15 so that the Business Analyst Lead can coordinate resources if applicable.

Each BA and Designer also writes an annual performance appraisal and submits it to the Business Analyst Lead for review. The Business Analyst Lead meets with each group member to review this annual appraisal and together they make any changes deemed necessary by the Business Analyst Lead before signing off and submitting to NAU Human Resources.

The business assistants are also assigned tasks for video transcription. Dana Stoneberger approves their timesheets. The business assistants manage the assignment of transcription tasks and process them to the status of Final Review.

6.1.1 Schedules and Records of Absence

While there is flexibility built into the BA and Designer roles, all members of the BA team are expected to work their required hours and attend scheduled meetings. Work and meetings, however, can often be fulfilled remotely depending on the nature of the work. The work of the BAs and Designer often require a mix of, 1) documentation, either via crafting designs or writing up

functional specifications, and 2) direct communication with project owners, Technical Team members, end users, etc. BAs and Designers both appreciate flexibility in regards to their schedules to account for both of these types of work.

When working remotely, BAs and Designers are expected to notify the Technical Team by email and note ways they can be reached. If a BA or Designer will miss work due to vacation or sick time, see below for business processes.

6.1.2 Sick Time

If BAs and the Designers will miss work due to illness, they are expected to notify the Technical Team by email. They are also expected to complete an online ROA and update their Outlook outgoing message accordingly.

6.1.3 Vacations

The Business Analyst Lead has asked that the BAs and the Designers document upcoming vacation days in their weekly 5-15s. BAs and Designers then submit an online ROA which is then reviewed for approval by the Business Analyst Lead. BAs and Designers are expected to notify the Technical Team of upcoming vacation days by email and update their Outlook outgoing message accordingly.

6.2 Design Processes

The design team fulfills two connected-but-unique roles in the development process at EC: I'll call them "functional design" and "aesthetic" design — which, along with Marketing-related design processes, fall under the umbrella of "user experience design." Our goal is always to advocate for the person using the tool, whether that is an employee of EC or a visitor to a public-facing site, and to ensure that their experience is a) as pleasant and frictionless as possible, while b) aligning their experience with our stated business goals. If we do our job well, there should be measurable benefit to the metrics which indicate the health and growth of EC's business and also a tangible sense of delight and ease by our internal staff and site visitors as they interact with our online tools and sites.

Practically, this means that a full-time designer and a senior developer will usually be involved early in the conversation with the BA and the client who has requested the development of a product. Based on input from the other parties, the designer will create a "Mockup" an initial sketch that shows the functional design and user interaction flow of a product's design. The client should be able to look at a mockup and have a clear sense of how they would use it and whether it will meet their needs.

Once the mockup is approved, the design team may opt to develop a "comp", a high-fidelity set of aesthetic designs produced in Photoshop or a similar tool, which shows what the final tool will look like on the page. The front-end developers can look at a Comp and a Mockup and extrapolate, essentially applying the aesthetic design of the Comp to the functional design of the Mockup.

Neither the Comp nor the Mockup are intended to be perfectly-final, pixel-perfect representations of the final developed product. Developers are expected to ask questions if they don't understand

something or if they feel something significant should be changed, but should have latitude to interpret these designs as needed in minor ways, to best address the problem at hand.

This is an art, not a science — there is of course a balance between (on one extreme) slavishly following the comp to the detriment of the product and (on the other) taking such liberties with the comp that it loses functionality or become more complex. A developer should especially ask hard questions before adding anything to the product that increases its complexity for the user.

Occasionally, the design team will make a functional prototype to test or communicate the tool's interaction and usability. In the past, we have used tools such as Axure, but increasingly we will be developing prototypes in the browser. Our eventual goal is to deliver these prototypes to the development team, who can then connect this shell and make it functional on the back end.

Finally, once a tool has been developed and is live on a test server, the design team will work with BAs and front-end developers to make sure that everything works as expected, and perhaps add some final polish to the CSS or user interaction.

The design team also serves the Marketing group, creating print or video materials which, unlike web development projects, are delivered directly to Marketing for approval.

6.3 Video Support Processes

Video Support Processes include:

- Live streaming for NAU-TV and other NAU entities (NAU Sports, ABOR, President, Graduation, ROTC and other by request);

- Video encoding (NAU-TV shows, EC Training, Disability Resources, Personalized Learning, others by request);

- Transcription of videos (NAU-TV shows, EC Training, Disability Resources, Personalized Learning others by request);

- Storage of videos for NAU-TV, Extended Campuses, Disability Resources and other NAU Entities

6.4 Project, Milestone, and Task Management

Effective management approaches for software development projects of new capabilities have evolved significantly over the last five decades. Our pursuit of an agile project management process in a rapid development environment has achieved its success either by design or by accident. In some cases, learning that our “out of the box” thinking was already a proven practice, and in other cases realizing the limitations of trying to work outside the box. Most of the problems we will encounter already have industry standard solutions. However, many require an investment in resources that exceed our budget or offer opportunities that we can use. By focusing on those issues with the most to gain and the least to lose, we can implement structure where we need it and stay flexible where we want it.

Three references provide an abundance of methodologies suited to the pursuit of solutions. “Agile Software Development Ecosystems” by Jim Highsmith (2002), “Kanban, Successful Evolutionary Changes for Your Technology Business” by David J. Anderson (2010), and “Agile Project Management” by Jim Highsmith (2013) are must reads for team members. We cannot implement every solution as described in these books. To achieve a balance between structure and flexibility, only those that can be beneficial and easily adapted to our flexible environment should be considered. Our flexibility has provided rapid response for fixing production issues and more reliable ways to development tools that meet the challenges of a constantly changing academic world.

However, management of the work requires tracking and accountability in order to report progress on a regular basis. The Task Management Environment (TME), developed by the team, provides ways to manage accountability and track the workflow. The TME, along with other processes and practices contribute to the roles and responsibilities of the programmers, engineers, designers, and business analyst.

BAs perform in the role of project managers managing all types of projects. (See paragraph 3: Technical Team Workflow diagram) The TME provides assistance to the BAs for managing projects. Projects differ based on who they support. Projects that support specific EC departments, other than the technical team, are considered external projects. External projects are typically identified as having a “New” or “Maintenance” cause. Internal projects may support other departments in general but not specifically because they are needed to maintain the infrastructure and administrative function of the technical team’s operations.

These internal projects support the technical team and have a cause of “Internal Administration.” TME internal administrative projects and tasks for internal administration can be created by any team member for training, supervising, and other administrative actions. TME internal maintenance projects and tasks are created and managed by the SASE group.

External projects also differ based on whether they are for maintenance of existing production capabilities or for the creation of new capabilities. These external projects are created and managed by the BA group. If an external project is for the purpose of addressing development of new capabilities, applications, or Web sites they are referred to as “development” projects and associated tasks are identified as having a cause of “New” in the TME. External projects for maintenance of current production capabilities are referred to as “maintenance” projects and associated tasks are identified as having a cause of “Maintenance” in the TME. These distinctions shall guide the creation of projects and management of tasks created to follow distinct processes.

Determining the scope and type of a project can be a daunting undertaking, but is a necessary step toward organizing the work to be done for a project. The first consideration for determining the scope of a project is independence from other projects. Overlapping scope for different projects should be avoided because it can lead to conflicting requirements. Interdependent functions should be grouped under one project. Also, separate processes for maintenance and development pose separate parameters for determination of scope. The phases and objective for development are different from the functional approach need for maintenance. Therefore, active projects in the TME

need to be organized to separate development projects from maintenance projects. At the time of completion, the status of a development project becomes inactive with its implementation in the production environment for use by the public or selected NAU/EC staff/faculty and (a) separate project(s) is (are) created for the maintenance of the new production capabilities. Another practice that can help guide the definition of scope is the use of the milestone feature in the TME.

Milestones can be used to subdivide projects to identify different functions or phases, depending on the type of task. Projects for maintenance of existing capabilities remain active for the life of the product. Project for development of new capabilities become inactive upon completion.

Making distinctions between development and maintenance projects is a standard practice in the software world. This is because there are major differences in the processes needed to ensure reliable applications development and the processes needed to ensure stability of a production environment during maintenance operations. The most significant difference has to do with the nature of the development process.

Software development of new capabilities involves the coding, debugging, unit testing, and integration of several tasks in order to produce a single functional component. Software maintenance of an existing capability, where the functional component already exists, normally involves a single task to correct or update the component. The only gray area with this decision between development and maintenance is when the modification to the existing capability will require multiple tasks. Major enhancements, requiring multiple tasks, should follow the development processes. Minor enhancements, needing only one task, can be done under the maintenance project as a maintenance task.

These tasks for new capabilities and major enhancements to existing capabilities are created with a cause of “New”, but are automatically changed to “Sprint” at the start of the next sprint. Progress on sprint tasks is reported on a real-time basis in the TME. Projects for ongoing maintenance of existing capabilities generate tasks with a cause of “Maintenance” in the TME and progress is also reported in the TME on a real-time basis.

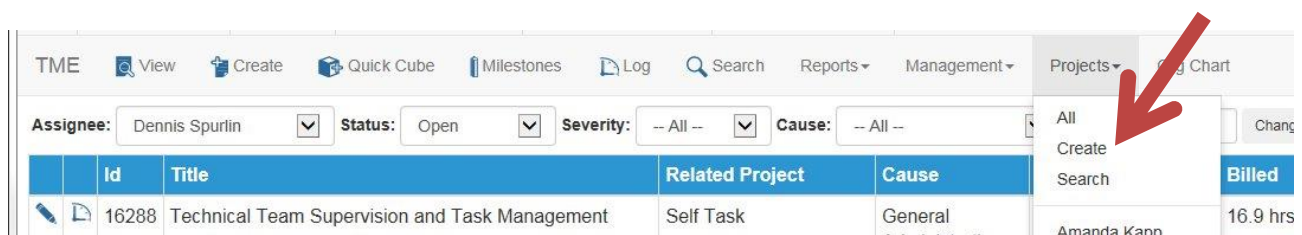
How to Use the TME

The TME is what we are using to document and manage projects, milestones, and tasks. It has been enhanced to the max and is in need of replacement. However, until we have sufficient resources to maintain our internal tools and processes we will have to continue to modify it for our use.

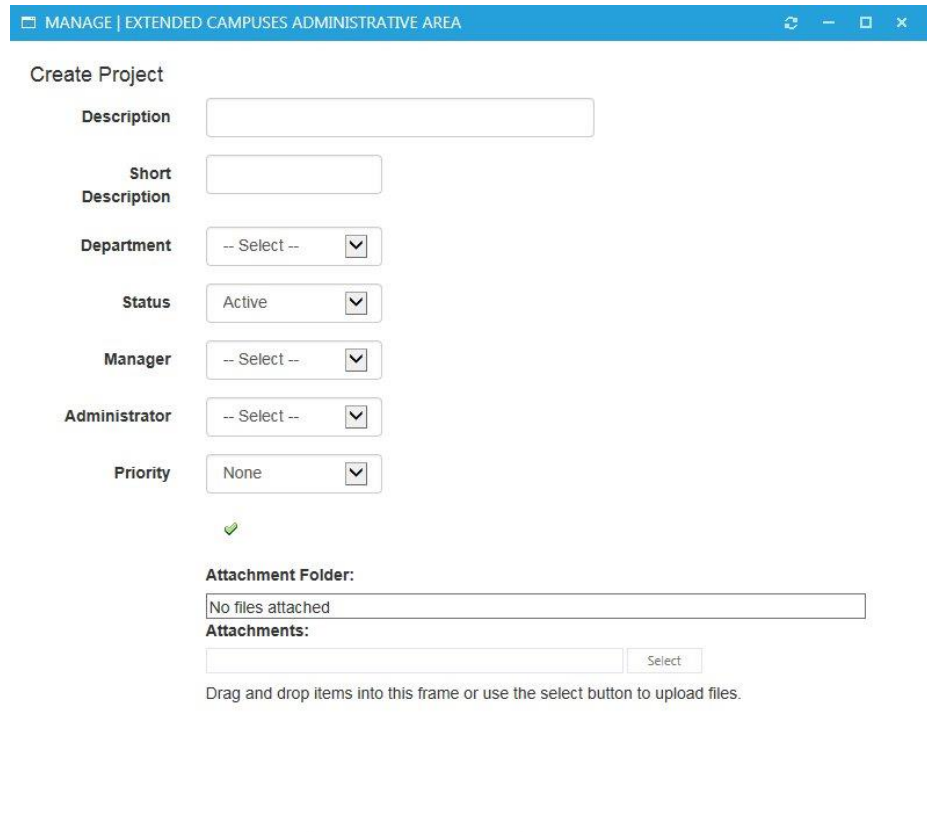
The TME is used by members of the BA group to create projects, milestones, and tasks for external customer support. Designers and transcriptionist in the BA group use it to document progress on assigned tasks. So let’s start with how it works for now when creating projects.

How to Create a Project

In the top navigation bar, select the Projects tab and then Create.



You will be greeted with this form. Fill out the form. Then select the green checkmark to create the project.



MANAGE | EXTENDED CAMPUSES ADMINISTRATIVE AREA

Create Project

Description

Short Description

Department -- Select --

Status Active

Manager -- Select --

Administrator -- Select --

Priority None

☒

Attachment Folder:

No files attached

Attachments:

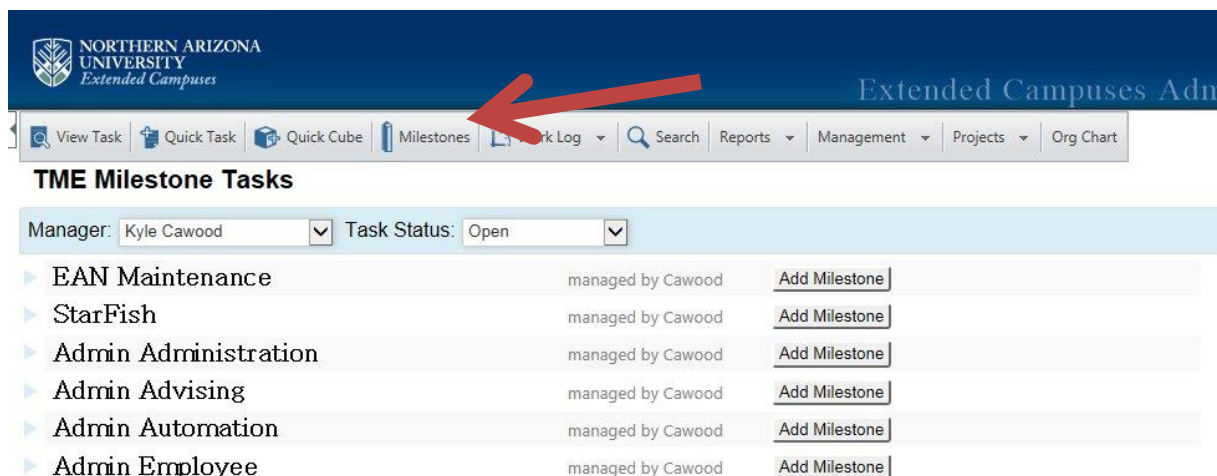
Drag and drop items into this frame or use the select button to upload files.

How to attach documents to a project:

1. Click the Select button for Attachments.
2. Navigate to the file in the pick list and double-click it.

How to Create a Milestone

In the top navigation bar, select the tab Milestones. By selecting Milestones, the Project Manager will be directed to a page, like the one below showing a complete list of their projects. To the right of every listed Project, is the button Add Milestone. Select this to create a Milestone for the prospective project.



NORTHERN ARIZONA UNIVERSITY
Extended Campuses

Extended Campuses Admin

View Task Quick Task Quick Cube Milestones Task Log Search Reports Management Projects Org Chart

TME Milestone Tasks

Manager: Kyle Cawood Task Status: Open

▶ EAN Maintenance	managed by Cawood	Add Milestone
▶ StarFish	managed by Cawood	Add Milestone
▶ Admin Administration	managed by Cawood	Add Milestone
▶ Admin Advising	managed by Cawood	Add Milestone
▶ Admin Automation	managed by Cawood	Add Milestone
▶ Admin Employee	managed by Cawood	Add Milestone

Complete the Add Milestone form and attach documentation specific to the milestone that will be common for all tasks created from the milestone by clicking the Attachments button. If the attachments folder does not automatically open when clicked, open \\dlsfileserver\Tme\Initial Review\Milestone_#### where #### is the milestone ID. Milestones may be edited after creation.

ADD MILESTONE CONTROL

Create New Milestone

Milestone Title: Add Enhancement for financial aid councilors

Description: Create new page with control for standard entry functions

Attachments

Required Date: -- Select --

Severity: 2- High * Department: Financial Support (E) * Requester: Billie Yazzie

Add Done

A project that has several milestones will expand to look like this:

QA for EC Redesign

managed by Oliver

Add Milestone

Complete Testing Documents	Add Item
Create Testing Documents	Add Item
<div>Edit</div> <div>Milestone Title</div> <div>Throw</div>	
<div> <div></div> <div>Create QA Doc - Admin Blog Tool</div> <div>Throw Task</div> </div>	
<div> <div></div> <div>Create QA Doc - Admin Events</div> <div>Throw Task</div> </div>	
<div> <div></div> <div>Create QA Doc - EC site - Campuses</div> <div>Throw Task</div> </div>	
<div>Edit</div> <div>Log</div> <div>ID</div> <div>Title</div> <div>Cause</div> <div>Due Date</div> <div>Est Hrs</div> <div>Billed</div> <div>Severity</div> <div>Status</div> <div>Assignee</div>	
<div> <div></div> <div>22214</div> <div>Create QA Doc - EC site - Widgets</div> <div>New</div> <div>11/29/2013</div> <div>8 hrs</div> <div>0.1 hrs</div> <div>3</div> <div>Initial Review</div> <div>Marc Lord</div> </div>	
Defect fixes from Testing	Add Item
QA Testing	Add Item
Tasks not related to a milestone	
<div>Edit</div> <div>Log</div> <div>ID</div> <div>Title</div> <div>Cause</div> <div>Due Date</div> <div>Est Hrs</div> <div>Billed</div> <div>Severity</div> <div>Status</div> <div>Assignee</div>	
<div> <div></div> <div>22213</div> <div>Collaboration with BAA testers</div> <div>New</div> <div>11/29/2013</div> <div>12 hrs</div> <div>0.1 hrs</div> <div>5</div> <div>Initial Review</div> <div>Marc Lord</div> </div>	

How to Add Items to a Milestone

Select “Add Item and enter a title in the popup. Adding an Item to a Milestone does not make that item a Task. An item uses all of the descriptive attributes of the milestone to create a task once the “Throw Task” button is clicked for the item.

QA for EC Redesign

managed by Oliver

Add Milestone

Complete Testing Documents	Add Item
Create Testing Documents	Add Item
<div>Edit</div> <div>Milestone Title</div> <div>Throw</div>	
<div> <div></div> <div>Create QA Doc - Admin Blog Tool</div> <div>Throw Task</div> </div>	
<div> <div></div> <div>Create QA Doc - Admin Events</div> <div>Throw Task</div> </div>	
<div> <div></div> <div>Create QA Doc - EC site - Campuse</div> <div>Throw Task</div> </div>	

ADD MILESTONE ITEM

Create New Milestone Item

Milestone Item Title

Add Done

How to Throw a Task from a Milestone

▼ QA for EC Redesign managed by Oliver Add Milestone		
▶ Complete Testing Documents Add Item		
▼ Create Testing Documents Add Item		
Edit	Milestone Title	Throw
	Create QA Doc - Admin Blog Tool	Throw Task
	Create QA Doc - Admin Events	Throw Task
	Create QA Doc - EC site - Campuses	Throw Task



Throwing a task will change the milestone item into a task and load it all filled out and ready for additional task specifications. Prior to the point of initial assignment to a full time member of the Technical Team, editing should be performed to include at this any additional attachments, descriptions, or updates to other attributes such as priority, requester, etc.

The screen capture below is an example of a task available for editing. The layout, order, and availability of fields and controls may change, but the attributes will remain constant. A brief description of each is listed below.

TME Task Options

Assignee: The default for this dropdown is the user creating the task. New tasks that will become sprint tasks may only be assigned to one full-time assignee by the project managers by selecting an assignee from the dropdown. Tasks for an internal administration cause may be assigned at the discretion of the creator so long as the tasks are for internal projects.

Attachments: The attachments option creates a folder located at \\dlsfileserver\Tme\Initial Review\Tme_nnnn. This folder is for any emails, query scripts, configuration files, diagrams, or documents related to the task. It should be used by both the project manager and the assignee.

Cause: The default cause is “New” upon creation of a task. The use of all available causes is explained below.

- **New:** Tasks for development of new capabilities, Web sites, or major enhancements are assigned with a cause of “New” and will be automatically changed to “Sprint” at the beginning of the next sprint. This is the default cause for tasks.
- **Sprint:** Sprint tasks are tracked in the manager’s reports to provide visibility to progress on projects. Tasks will automatically be changed to sprint tasks when appropriate. Users are never to select a cause of sprint.
- **Internal Administration:** Tasks may be created by any team member, at their discretion, for the purpose of tracking and accounting for administrative responsibilities. Internal maintenance tasks created for projects managed by the SASE group also use this cause. This cause is not set by default and must be manually selected.

- **Maintenance:** This cause is available for maintenance of production capabilities for projects managed by the BA group. This cause is not set by default and must be manually selected.

TME View Create Quick Cube Milestones Log Search Reports Management Projects Org Chart

t27830: Create Course Approvals Control

Copy Title Cancel Update

Title:
Create Course Approvals Control

Status: Assigned Severity: 2- High Task Phase: Development *Department: Academic Operations (EO)

Required Date: --Select-- Due Date: 10/20/2014 Start Date: 10/6/2014 *Requester: Kyle Cawood *Cause: New

*Project Leader: Kyle Cawood Schedule Date/Time: 10/6/2014 3:14:23 PM URL: [Open Page](#)

Assignee:
Adam Cook
Adam Thomas
Alec Kantor
Amanda Kapp
Andrew Adams
Bee Ronan
Blanche Johnson
Christopher Wong
Damien Coy
Daniel Garcia Briseno
David Brink
David Dennehey
Dennis Spurlin
Donald Lacopo

Description:
Enter detailed task description

Task Attachments

Milestone Description: None

Milestone: Incorporate Instructor Approval into Milestone Attachment: Milestone \\dfs\server\Tme\Initial Review\Milestone_137 Contents: Contents

*Project: Admin Faculty Management Project Attachments: Projects No files attached to project Contents: Contents

Creator: Kyle Cawood Entry Date: 10/6/2014 3:14:14 PM *Estimated Duration: 8 Work Type: Technical Team
Actual: 0.20 hrs, Percent 2.50 %

Cancel Update

Work Logged (Total Hours: 0.2)

Date Worked	User	Hours	Description
10/06/2014	knc7	0.1	Assignee(s) changed from Kyle Cawood to Joseph Fileger. Task status changed from Initial Review to Assigned.
10/06/2014	knc7	0.1	Created task.

Creator: The default for this field is the user creating the task and cannot be altered by the TME.

Department: This is also a required field. It may be automatically populated from a milestone item, but it must be entered when using the Quick Task button. Select the department best supported by the task from the following list. Do not select the Technical Team for external customers.

Description: The description field is for describing the requirements and scope of the task. Details may also be documented in the attachments.

Due Date: The default is the date the task was created plus fourteen days. It should be updated for another fourteen days when the task is assigned to a full-time team member to be worked.

Entry Date: The default for this field is the date of creation and cannot be altered by the TME.

Est. Cost: Traditionally, the unit used for estimating tasks costs has been hours. A precision of four hours is sufficient. Estimates in 4 hour increments should be approached from a perspective of assurance that the task will take less than the estimate. Sprint tasks are to be limited in size so as not to overload the assignee. Therefore, the recommended options for estimated cost of tasks with a cause of “New” and “Sprint” are 4, 8, 12, 16, and 20. The preservation of initial estimates provides essential metrics for the improvement of estimates.

Milestone (dropdown): This is automatically populated from a milestone when a task is thrown from a milestone item. When the task is created from the Quick Task button, there will be no item in the dropdown unless the project has milestones. Milestones are editable.

Milestone (button): The milestone button is available to access the folder where milestone documentation may have been place when the milestone was created.

Project (dropdown): The project dropdown is a required field that may be inherited from a milestone or selected by the creator at task creation.

Project (button): The project button is available to access the folder where project documentation may have been placed when the project was created.

Project Leader: This field is automatically populated when the project is selected. It should not be changed without the project manager’s coordination.

Requester: The requester defaults to the creator of the task either by milestone or Quick Task button. An email notification is sent to the requester to inform them of the task number for reference. This field should be changed when the person who should get the email is not the creator of the task. The list of options is taken from the Admin Users table and includes all of EC.

Required Date: This is an optional field. This is only needed when there is a “Drop Dead” date on the task. In other words, if the task is not done by this date, it will have been futile. If the work will have useful results if it is complete past the due date, the required date is not needed.

Severity: Be sure to update as needed. This is different than the project’s priority. Priority carries an assessment by management with regard to development projects. Severity is the Technical Team’s assessment of the operational impact of a task. This assessment can be made by considering two separate functions of the Technical Team, maintenance and development.

Maintenance: Special considerations for existing production capabilities include whether or not there is a work-around for the problem and whether the problem has limited function, just doesn't work at all, or causes compounded dysfunction. The two conditions for work-around should be combined with three conditions for functionality. The worst kind of functionality would be where use by a customer has adverse effects to cause compounded dysfunctions. Limited function would have less impact than no function. So the following table can provide a starting point for determining the severity for a maintenance task.

Maintenance Issue Severity

Production Capabilities Work-Around	Functionality	Severity
No	Adverse	1
Yes	Adverse	1
No	None	1 or 2
No	Limited	2
Yes	None	2 or 3
Yes	Limited	3
None needed	Minor function change	3 or 4
None needed	Minor interface change	4 or 5

Development: Severity is used to sequence the order in which development tasks for a specific milestone are to be worked. The available values range from 1 to 255 (tinyint).

Start Date: The default is the date the task was created. It is updated to the date the task is assigned to a full-time team member to be worked upon task assignment.

Status: The initial status of tasks upon creation is "Initial Review." The normal flow for changes in status goes from "initial review" to "assigned", then to "in-progress", then to "in-testing", then to "final review", and finally to "closed". Other status' are available and should be used when appropriate.

- **Initial Review:** This status is for initial evaluation of prioritization, sizing, and requirements prior to assigning the task.
- **Assigned:** Tasks in "Assigned" status have been assigned for work to begin, but have yet to be worked on.
- **In Progress:** It is the responsibility of the assignee selected to work the task to change the status to "In-Progress" once they have begun work on it.
- **In Testing:** In the case of maintenance, it is the responsibility of the assignee, to change the status to "In-Testing" once they have submitted the task for acceptance testing. In the case of development, it is the responsibility of the assignee, to change the status to "In-Testing" once they have submitted the task for integration testing.
- **Final Review:** In the case of maintenance, it is the responsibility of the assignee, to change the status to "Final Review" once they have verified that the task has been deployed to the production environment. In the case of development, it is the responsibility of the assignee, to change the status to "Final Review" once integration testing proves successful.

- Closed: Only a task manager is responsible for closing new, sprint, and maintenance tasks. Other team members may close only the internal administrative tasks they have assigned to themselves.
- Canceled: Only the project manager for a task may change the status to “Canceled.”
- On Hold: Only the project manager for a task may change the status to “On-Hold.”
- Backlog: Used by project managers to temporarily hide tasks from lists.
- Queued: Used by project managers to queue tasks ready to be worked by selected developers.
- Ongoing: This status only used for internal administrative task.

Task Phase: The task phase attribute is a list of categories that can be used to catalog tasks into groups that can be measured separately. The purpose is to gain metrics to improve task estimation methodologies. Studies of these metrics, over time, are expected to yield insight into a better list of attributes. Ongoing refinement and study of this task phase attribute depends on improved task specification and analysis to decompose the categories more distinctly.

Title: The title is a required field. It may be automatically populated from a milestone item, but it must be entered when using the Quick Task button. The title provides a distinctive way to reference a task without having to use the task number. It is not a description of the task and should fit easily in the field provided.

URL: This is an optional field where a link to a Web page may be inserted. The Web page may be on one of our sites where there is an error, or it may be to a location for additional information on the task.

Work Logged: The quarter-clock icon allows the assignee working the task to log details of their work with the time invested with each session of work on the task. It is the responsibility of the assignee to maintain work logs on a daily basis. These log entries are displayed at the bottom of the task form for the benefit of task and project and managers. Instructions for assignee use of the TME to include work log is documented elsewhere.

Work Type: The default work type is “Technical Team.” The computer support staff uses the TME with a different configuration. Do not change the work type when creating or processing a task.

TME Menu Options

Search: There are two types of Search. First there is the basic search by task number or by key word in the title. Then there is the advanced search.

Clicking the Advanced Search link on the basic search page provides options to search by specific task attributes.



TME Search

Work Type:	-- Select --	
Task #		
Title:		
Description:		
Work Log:		
Tasks created after:		
Tasks created before:	1/21/2014	
Assignee:	-- Select --	
Requester:	-- Select --	
Project Manager:	-- Select --	
Project:	-- Select --	Include Inactive? <input type="checkbox"/>
Department:	-- Select --	
Status:	-- Select --	
Cause:	-- Select --	
Phase:	-- Select --	
<input type="button" value="Search"/> Basic Search		

Reports: There are currently five reports available from the reports menu option in the TME.

- The 515 Report will list all work log entries by date for a selected period of time for selected team members. The default is for the current week and the user.
- The Sprint Report lists all active sprint tasks by assignee and status. Overdue and languishing task are indicated with red highlights on the dates.
- The Load Report lists remaining hours for all tasks and a summary of total load for each assignee.
- The Sprint Progress Report provides a list of tasks complete and in Final Review by each assignee. It also provides a date for their last work log entry.
- The Managers Report is reviewed at the end of each sprint. It lists task by project manager and project.

Management: Provides an option for supervisors to manage Capacities.

Projects: This menu option is used to search project managers, manage projects and milestones, and create new projects. The dropdown lists all of those assigned as project managers to existing projects. Selecting a project manager will open a page with a list of all the projects assigned to the selected project manager. Clicking “All” will show a list of all the projects for each project manager with the active projects listed first. Selecting the search option will offer a choice for listing only active, only inactive, or all projects based on key words located in the description and short description of the projects.



Projects displayed based on the selection made from this list will have a Milestones button to the right of the edit pencil. You may edit a project by clicking the pencil or examine its milestones by clicking the milestone button.

Project Search : Project Status: Active									
Edit	View Milestones	Manager	Description	Short Description	Department	Link	Administrator	Owner	Priority
	<input type="button" value="Milestones"/>	Dennis Spurlin	Admin – Computer Support and Task Management Environment menu items	Admin TME	Technical Team (EC)	\\dlsfileserver\Tme\Initial Review\Project_88	Dennis Spurlin	Marc Lord	0

Selecting the Milestones button will open a page listing the milestones for the associated project.

Milestones for Project 88

Milestone_ID	Status	Title	Description	Severity	Attachment_Location	Required_Date
34	Active	TME Usability and Interface Enhancements	Use the latest toolsets to enhance the TME and improve the usability.	3	\\dlsfileserver\Tme\Initial Review\Milestone_34	
35	Active	Add the concept of Task Monitor to the TME	The Task Monitor is an alternate project manager that needs to be able to create milestones and tasks for a project managed by another project manager. Projects shall appear in the list for management of milestone for both the project manager and the task monitor. Tasks shall appear in the View Task list at the bottom in blue for both the project manager and the task monitor. All access and defaults for a project shall be the same for both the project manager and the task monitor	3		

Clicking any of the milestones in the list will open the Milestone page to the milestone selected.

TME Milestone Tasks

Manager:
Project Status:
Task Status:

Admin TME managed by Spurlin [Add Milestone](#)

35 Add the concept of Task Monitor to the TME [Add Item](#)

Edit	Milestone Title	Throw
	Add column for Task_Monitor to the Tme_Task table	Throw Task
	Add column for Task_Monitor to the Tme_Projects table	Throw Task
	Include TME projects on milestones for task monitor	Throw Task
	Include tasks for task monitor in TME view task	Throw Task

34 TME Usability and Interface Enhancements [Add Item](#)

0 Tasks not related to a milestone

The Project menu item is a handy way to manage projects and milestones.

7. System Architecture and Software Engineers group Roles and Responsibilities

The System Architecture and Software Engineers (SASE) group, led by Kevin Hayes, Lead Applications Systems Analyst, Programmer, is responsible for server systems administration, database server management, and software systems architecture for Extended Campuses and NAU TV. Maintenance and development projects for external customers are managed by members of the BA group. Projects for maintenance and upgrading the infrastructure comprised of the database

servers, Web servers, file servers, content management systems, and software architecture, development processes, and automated tools are managed by members of the SASE group.

Each member of the SASE group possesses a wide range of skills. Roles and responsibilities are distinctly assigned to each member of the SASE group with sufficient overlap in skills and knowledge to provide backup while one is unavailable. The members of the SASE group are: Kevin Hayes (Applications Systems Analyst, Programmer Lead), Marc Colvin (Computer Database Administrator), David Brink (Computer Database Administrator), John Meyer (Systems Administrator Senior), Vivek Bongu (Applications Systems Analyst, Lead), and Dave Dennehey (Applications Systems Analyst, Lead). Dennis Spurlin (Applications Systems Analyst, Programmer Senior) also manages projects for the maintenance and upgrading of the internal infrastructure and support systems for the Technical Team. SASE members are not limited to project management nor are they limited to the roles and responsibilities listed below. The roles and responsibilities described in the next four subparagraphs describe primary responsibilities for roles that are likely to also be filled by another in their absence.

7.1 SASE Management

Kevin Hayes: Kevin Hayes is an ASA programmer team lead and supervises the SASE Group of two database administrators, one senior systems administrator and two ASA team leads

7.2 Database Management

Marc Colvin: Manages all databases and SQL servers housed by Extended Campuses. He has built and maintains nearly every process that pulls and transforms data from other systems on campus such as Peoplesoft which is then used by all of our public and private facing web sites. There are many different technologies Marc uses that keep our system in sync with Peoplesoft and other university systems. Some of the technologies include SQL server Integration Services, file generation and pull through SFTP, accessing web services, LDAP and custom C# applications and services that do tasks such as monitoring all of our systems, send out batch emails and run data integrity checks. In addition to data syncing and transformation Marc also has many complex processes built within SQL Server in place to do transcript processing, report generation, and connect integration. Optimization of queries, helping others write SQL and building data structure and design for new projects also falls under Marc's umbrella on a regular basis. Marc acts as a Backup for David Brink and assists with Business Intelligence processes as needed.

David Brink: Manages the data warehouse and Business Intelligence (BI) processes that support Business Objects and other reporting systems. This involves installing, configuring, and maintaining the BI database servers and other servers used for reporting, writing the Extraction, Transformation, and Loading (ETL) scripts, SQL Server Integration Services (SSIS) packages, and applications/utilities necessary to provide our reports with the data required by our users. He is also backup for Marc Colvin and John Meyer.

7.3 Server Systems Management

John Meyer: The role of Systems Administer Sr. is to assume responsibility for the planning, provisioning, configuring, maintaining and supporting the stability and reliability of the following: power distribution; uninterruptible power supplies; physical servers; virtual servers; all server software packages and licensing; firewalls, IPS modules, and general network security; network infrastructure over both fiber and copper; storage area network (SAN); data retention and backups; SSL certificates; Active Directory group policy and group based access control. Many of these responsibilities require working closely with staff at NAU ITS, primarily the IS, SIA, NOC, and MENSA teams.

The role also serves as front-line support for several environments, including the following: SharePoint, fileserver, and OnBase access control; Wowza live and video-on-demand streaming; XenDesktop and associated thin clients; Picturepark digital asset management for EC Marketing; CatDV digital asset management for Television Services; all servers, embedded hardware, and software packages involved in the operations of Television Services in Master Control as well as the mobile production truck, including Grass Valley, Playbox, BrightSign, and others. Additionally, the role provides Tier 2/3 end user support to the Technical Support Coordinators.

7.4 Software Architecture Management

Vivek Bongu: In the role of the Technical Team's chief software engineer, he consults with BAs on project analysis and task requirements, with the ASA senior (Damien Coy) of the development cadre on software architectural issues and standards, and with the ASA lead maintenance chief (Dave Dennehey) on software systems sustainment issues related to deployment and maintenance of new development.

Dave Dennehey: In the role of the Technical Team's maintenance chief, he assumes responsibility for all maintenance of existing EC Software. To assist with this responsibility, he has primary access to the apprentice programmers and ASAs of the maintenance cadre and reviews their work to ensure it meets standards. He manages the use of apprentice programmers by fulltime programmers of the development cadre and coordinates the use of additional resources from the development cadre with the BAs when needed. He reviews training and documentation of instructions and guidance for apprentice programmers and new ASAs. He coordinates all upgrades and configuration changes for Ektron, the integration and deployment processes, and the architecture that supports the development and maintenance of software systems.

The role of maintenance chief deals primarily with all EC.Solutions requests for maintenance to be queued and routed by the BAs for assignment by the maintenance chief to members of the maintenance cadre. Maintenance is defined as taking software from a broken state to a working state, not as major enhancements that extend the functionality to provide new capabilities. It includes upgrades to software libraries and third-party tools such as Telerik and CC.net, the file structure of version control systems and Web sites, and the configuration of Web-facing services and servers. The maintenance chief also performs as backup to John Meyer for SharePoint and Server Administration, to Dennis Spurlin for administration and supervision of the programming

group comprised of the development and maintenance cadres, and to Vivek Bongu for project analysis consultations with the BAs.

Dennis Spurlin: Reports directly to the Technical Director. He supervises the programming group comprised of the development cadre and the maintenance cadre. He is responsible for conducting scheduled reviews and appraisals of performance, accounting for missed time, reporting work performed, and ensuring that members of the programming group follow NAU policies. He also functions as task manager by monitoring the flow of tasks for development and maintenance, collecting and reporting metrics on individual and team productivity, and coordinating collaboration for process improvement proposals. He represents the programming group as an ancillary member of the SASE group and as backup to the maintenance chief.

7.5 Internal Project Management

As explained in paragraph 4.2.1, internal projects have no specific external customer other than the technical team. Projects for the administration of hardware and software systems, supporting the team's ability to provide applications and tools for use by those outside of the technical team, are managed as internal projects. Maintenance and development projects for external customers are managed by members of the BA group. Projects for internal maintenance and development are managed by members of the SASE group. Tasks created for internal projects have a cause of "Internal Administration."

7.6 Resource Management

The SASE team submits a weekly 515 report to their supervisor due by close of business on Friday of each week. The 515 report can be automatically generated using the TME, but this is not required. Any work a SASE member does toward a specific task or project is tracked and stored within the TME. Given that the SASE group functions very fluidly responding to a variety of problems and new technical challenges, the SASE group logs work in the TME differently from the rest of the Technical Team. SASE members can have ongoing tasks which they use to log their daily work. These log entries can be put into an automated report and emailed to the SASE Manager and Technical Team Director, where it is further decomposed and forwarded to EC Administration. SASE members can also manually generate their 515's. SASE members may also log work directly to tasks assigned to them- however it is important to note that these tasks are not directly reported on from a SASE perspective (unless added to their daily work or put in their 515's). The SASE team follows NAU guidelines, policies and procedures in the use of its Records of Absence and schedules.