Managing a GIS Enabled Web Application from Requirements to Deployment

Sandrine Schultz (sandrine.schultz@navy.mil)
Ryan Taylor (@ryanewtaylor)
Navy Shore Geospatial Energy Module
Application Architecture

Part of the GeoReadiness Enterprise System (GES)

ArcGIS Server 10.1
Oracle 11g
ASP.NET & ASP.NET MVC
Apache Flex
Crystal Reports Runtime
Sphinx
Statistics

4th year in development
1 year contracts
1 new version released every month
2,333 task tracking tickets to date
5,950+ commits to source control
≈ 700 code reviews
≈ 1,000 users
Challenges

Capturing & prioritizing requirements
Being responsive to user feedback & needs
Adapting to changing needs and new info
Communicating with stakeholders
Communicating with a distributed team
Understanding status & progress of all things
Our Solution(s)
Waterfall Development

Too many unknowns
Requirements naturally change over time
Long term predictions become difficult
Embrace change rather than manage it away
Iterative Development

Break development cycle into small sprints
Reduce requirements to deployment timespan
Increase feedback frequency from users
Improve ability to respond to changes
Sprint Logistics

1 sprint = 6 weeks =
  4 weeks development & testing
  2 weeks deployment & testing
Prior sprint overlaps next sprint by 2 weeks
New version deployed every 4 weeks (or so)
Sprint Logistics in Graph Form
Sprint Planning
Active Sprint
Sprint Activities

Planning (pre-sprint)
Requirements
Daily huddles
Design
Development
Testing
Deployment
Planning

Big Picture Planning
Start of fiscal year
Review prior year effort
Establish success criteria
ROM for big items
Rough schedule of events
Whole team

Pre-Sprint Planning
Beginning of each sprint
Review backlog for high priority tickets
Populate sprint
Review with whole team
Requirements (continuous)

Captured Via Several Mechanisms
Feedback forums
Steering groups
Help desk
Internal suggestions

All requirements become J IRA issues for tracking history, progress, and status
Tracking Requirements

Our workflow tracks status of all features, enhancements, bugs, and tasks
Tracking Requirements
Daily Huddles

Foster team collaboration \ communication

What were you working on?
What are you working on today?
What roadblocks do you have?

Tuesday question of the day is fun!
Development (Identifying Tasks)
Development (Committing Work)
Development (Code Reviews)

Pull request

#675  OPEN  NSGEM-5033 → NSGEM-4810

NSGEM-5033 return available reports and their

using System;
public class ParameterDataSource
{
    public string Url { get; set; }
    public string ValueField { get; set; }
    public DataType DataType { get; set; }

    public ParameterDataSource(string url, string valueField, string dataType)
    {
        Url = url;
    }

    Ryan Taylor
    I think we should pass in the DataType enumeration here rather than the string. This will enforce, at compile time, only valid DataTypes for the constructor.
    Reply  Edit  Delete  Create task  2 days ago
Development (Gitflow)

Image Credit: http://bit.ly/1xTAXJS
Development (Gitflow)
Development (Automation)

On Every Commit
New build created & deployed internally
External deployment packages created
J IRA issue statuses updated automatically

Repeatable, predictable, consistent
Saves a lot of time
Testing

JIRA issue describes success criteria
Features / bugs tested as they’re completed
Reopen JIRA issue if test fails
History captured as comments in issue

### Description
To Test

1. Launch NSGEM
2. Select **CND Washington** from the **Region** lens
3. Select **N33355** from the **Installation** lens
4. Scroll down to the **CIRCUITS Installation/Site Consumption and Reduction by Type Chart**
5. Hover over the slider bar located at the edge of the chart and drag the bottom scroll bar to the x-axis of the chart
6. Generate a custom report containing the **CIRCUITS Installation/Site Consumption and Reduction by Type Chart**
7. Compare the graphs

### Expected Result
The custom report and dashboard should display the same number of facility types.
Deployment

Feature Freeze Day

Once per sprint
All features must be complete & tested
Create release branch
Fix remaining issues (if any)
Development team starts working on next version
Deployment packages automatically built
Promotion team promotes app through 3 environs
Fully tested in each environment
Release complete when all tickets pass
Iterate

Requirements → Design → Development → Testing → Planning → Deployment

Continuously reflect on your processes and tools in order to improve upon them.
For More Info

cnic.navy.mil

Ms. Sandrine Schultz, CEM
CNIC, Energy Program Manager
DSN: 288-6293 COMM: 202-433-6293
E-Mail: sandrine.schultz@navy.mil