* HTML5 Pilot Project
  + Problem:
    - Our current website, Dolv3, is running on the Silverlight platform. Silverlight will be discontinued by major web browsers in the near future, requiring us to build a replacement.
  + Solution:
    - Development of prototype website utilizing modern web technologies. We decided to use the MEAN stack supplemented by Windows microservices.
  + Result:
    - Improved load time of corresponding pages from 4 seconds to under 300 milliseconds.
    - Enhanced user experience though modernization of user interface.
    - Produced quality code in line with current best practices, encouraging code readability, reuse, and extensibility.
    - Validated code correctness through array of unit and integration tests.
    - Established continuous integration environment to build and test code upon pull requests.
* Token Authentication Service
  + Problem:
    - Users on the prototype replacement for Dolv3 require secure authentication to access the site
  + Solution:
    - Development of token-based authentication microservice using ASP.NET WebAPI and MongoDB
  + Result:
    - Improved security over basic authentication by negating the need to send user credentials through the request header
    - Achieved authentication request time of approximately 60 milliseconds
    - Provided automatic token invalidation through time and IP based checks
    - Facilitated customization of token rules through minimal and complete interface
    - Designed basic safeguards to mitigate malicious logins by counting failed attempts
* User Data Service
  + Problem:
    - User information stored in SQL servers must be made available to several parts of our prototype website
  + Solution
    - Development of REST microservice to allow others services to request information about specific users
  + Result
    - Allowed services to retrieve user data without making SQL calls
    - Enabled system to serve data only belonging to that user, ensuring data privacy
* Asset Summary Service
  + Problem:
    - Entire asset summary is periodically being cached into MongoDB from SQL, but services only require specific parts of result set
  + Solution
    - Development of microservice to read through stored asset summary and retrieve records correlated with the given identification key
  + Result
    - Reduced data retrieval time of 2200 records from approximately 4 seconds to under 400 milliseconds compared to Silverlight system
    - Improved load time of user interface by serving preformatted data
* DOL Rest Service
  + Problem:
    - Having several server-side microservices on different routes complicates REST calls from the client
  + Solution:
    - Development of microservice which encapsulates server-side services and exposes a single interface
  + Result:
    - Simplified creation of REST calls to the backend through a coherent interface, leading to simpler and more consistent code
    - Prevented unauthorized access to microservices by only publicizing routes needed by the client
    - Enhanced logging features through centralization of server traffic
    - Allowed prototype application to operate over a single firewall port, improving system security
    - Provided functionality to combine REST calls into a single route, allowing the client to do less work
* Mongo Access Layer
  + Problem:
    - Services require an abstraction through which to interact with MongoDB
  + Solution:
    - Development of data access layer allowing services to perform CRUD operations with their business objects
  + Result:
    - Reduced coupling with the database by separating implementation of the database from business logic
    - Promoted code reusability by engineering solution with generic type arguments
    - Enabled modification of database operations without impacting the business logic code
    - Bolstered functionality of CRUD operations by supporting LINQ queries in function arguments
* Baidu Maps
  + Problem:
    - Customers in China are unable to access our Google Maps application due to government restrictions
  + Solution:
    - Development of Baidu Maps clone of our Google Maps application
  + Results:
    - Achieved core functionality on the Baidu platform, enabling blocked customers utilize our mapping data
    - Cleaned up code before writing new features, reducing pollution of the global namespace, improving readability, reliability and performance
    - Organized mapping code into duck-typed interface, accommodating Baidu without needing to rewrite underlying business logic
  + Microservices:
    - Asset Summary REST Service
      * Retrieves assets/datachannels from Mongo
    - DOL REST Service
      * Public interface(API?) to access protected REST services
    - User Auth Service
      * Custom token-based authentication using Mongo
    - User Data Service
      * Retrieves data pertaining to user
  + Libraries:
    - Authentication Attribute
      * C# Attribute for requiring authentication on MVC class
    - Basic Authentication
      * Authenticates user by comparing password in header
    - REST Client Library
      * Interface for making synchronous REST calls
  + Other:
    - Unit/Integration tests for every service and library
    - Continuous integration environment
    - Consistent logging at multiple levels
    - Custom exceptions and unified handling
* Packet Counter Service
  + Problem:
    - Counting carrier packets on the Dolv3 web client created excessive database load, leading to slow page load time and reduced database performance.
  + Solution:
    - Creation of microservice to perform packet counting in memory and store the count in the database
  + Result:
    - Packet count webpage load time decreased from 30 seconds to under one second
    - Eliminated over 2 million database operations per page instance per minute, significantly reducing database strain leading to system-wide performance increases
    - Improved code quality by moving business logic out of the user interface and database
* Google Maps
  + Cleanup of core maps code
    - Reduced pollution of global namespace
    - Moved business logic into duck-typed interface
    - Organized code through prototypes and local data structures
  + Addition of Baidu Maps
    - Implemented interface with Baidu Maps API
    - Allow Chinese customers unable to use Google access to our mapping system
  + Maps Overhaul
    - Cleaned up Maps
      * Reduced number of user states
      * Simplified presentation of data
      * Simplified navigation
      * Enforced consistency
* Build Process Enhancements
  + Build Script
    - Split monolithic build script into functions for easier modification and readability
    - Modified process to execute build script from pulled source as opposed to static location
    - Helped parallelize build script, reducing build time by about 40 minutes
  + Publish Script
    - Created script to automatically deploy published VS2013 solutions to IIS
* WiX
  + Independently learned and implemented WiX installer for our Visual Studio solutions
  + Included custom scripts to gather required files for packaging into installer
  + Improved speed and reliability for installation of services
  + Wrote documentation detailing process for creation
* Billing
  + Created program to automatically process customer billing records
  + Improved productivity of HR by automating tedious process
* Design Curve
  + Created data visualization for helium iso container showing current pressure against tank-specific pressure limit curve
  + Aided customers in determining the health of their assets while in transit
  + Offered insight into determining which container type is most efficient