MN.IT Services @ DOT, Enterprise Solution Group

Technical Development Architecture Guide
GIS Software Development

Version 10.0.5
January 3, 2013

Authored by: Charlie McCarty, MN.IT Services @ DOT Enterprise Services Development Unit - GIS
Architect and Unit Supervisor
Charlie.mccarty@state.mn.us
1. Introduction

This document describes the general software development environment to be used for GIS projects at MnDOT. It is the basic environment that is used at MnDOT using the ArcGIS version 10.0 technologies. At this writing, MnDOT has not yet gone into the ArcGIS 10.1 environment, but anticipates it will do so in the server, services, and applications side during the first part of the 2nd quarter of 2013.

Tailored and more specific versions of this document can be developed to fit project needs, during project elaboration phases.

This document applies to software development projects for or integrating with desktop ArcGIS, ArcGIS Geoprocessing, ArcGIS Server applications, GIS based web services, and mobile GIS/GPS development at MnDOT.

MnDOT GIS projects are developed primarily in Microsoft .NET environment. There exist mobile environments and integration efforts that require several different development environments.

Although there is the ability to develop in server side Java or Asp.Net with ArcGIS ADF, MnDOT will no longer do any ArcGIS ADF based projects. ArcIMS is also a technology MnDOT no longer works with going forward.

Increasingly, ArcGIS Rest API, ArcGIS Flex API, ArcGIS Silverlight API, ArcGIS JavaScript API, and supporting technologies such as HTML5 (IE 9.0, Chrome, etc), .Net (asp.net, wcf, etc), Android OS, and iOS are being worked with in the MnDOT GIS development environment. Consult with project managers and project architects about where these transitions are.

The supported data and system architecture for GIS projects are described in this document. If you do not see a particular database, application server environment, or development framework mentioned here, it is likely not supported and would require extensive and timely change management processes to consider using it.
2. Benefits

The following benefits are expected from this development architecture and its description.

- Conforms to an environment that is easy for MnDOT OI&TS’ Infrastructure Services to support.
- Allows for the proper handling of GIS data from ArcGIS Services.
- Conforms to the main GIS software used at MnDOT – ESRI’s ArcGIS Suite (Enterprise Levels).
- Is object-oriented, which promotes component-based, reusable, and extendable code. This makes it relatively easy to change, add functionality, and interface to other systems/services.
- Is service-oriented, which promotes the use of common services such as directory-based authentication, standard map services, location services (as web services), and special application specific services.
- Is cost effective when considering 5-10 development projects per year, over several years that these related technologies will be supported.
- Low entry cost for developers outside of OI&TS-APPDEVGIS
- A robust and well exercised development environment accrues benefits towards solutions building.

3. GIS Architectural Overview

Development/Test/Production Server Configuration

MnDOT Development, Test, and Production servers are now virtually served, using VMware software and large shared data center technologies.

The following virtual GIS Server Development environment will be the environment that projects are initially deployed to. The TEST and PRODUCTION versions are the same.

The database tier is represented by the Oracle TGD instance. Schema changes using developer accounts are allowed on TGD, and once the schema is understood, it is transferred to application accounts on TGD where no schema changes are allowed, only CRUD changes to data. This same transfer then will occur to test and production data environments, where only CRUD changes are allowed. Data Management tools or administrative tools are almost always needed to keep most applications data up to date, and they should be planned for.
Note that two of the three Application SOM/SOC machines (at top level) are network load balanced machines and these are the primary targets of deployment. One SOM/SOC machine is not load balanced and is in place for legacy applications that do not perform well in a load balanced environment. All development should be designed for load balancing – and this means NO LOCAL DATA IS STORED on the application servers. All data is stored in an enterprise database separate from these machines.

Direct access is not given to the ArcGIS Image servers – any image or raster aspects of development must be handled by MnDOT staff for deployments.

GIS Files, File Geodatabase, and FME conversion processes can run on a provided server, however, the output is intended to be published to enterprise databases, GIS files and File Geodatabase structures are not to be used by the application servers and ArcGIS services directly.
Developers are not granted access or control to TEST and PRODUCTION server environments. All applications developed must port and install to these environments without any help needed from the developer directly.

All development must be deployable (installable and configurable) with the minimum amount of effort to ALL levels of server environment.

A file system structure of where applications’ files reside exists, and is presented in another document available at project start. Logfiles, web site, shared map caches, web input and output directories are made clear to each project developer.

Beyond development, all subsequent data and application management tasks must be thought out and a management console or tool and process provided to perform management tasks at the TEST and PRODUCTION levels. There is no direct editing of supporting database tables at these higher levels, it must be done via application or well understood and related COTS software available to MnDOT Data and application stewards. There are no direct configurations of applications at the TEST and PRODUCTION levels...a well understood process and software tool must do this management, securely.

**Application Technology**
The application development / maintenance technology for ArcGIS Services applications is as follows:

**Application architecture:** Microsoft .Net 4.0, ArcGIS Server 10.0 sp5. ESRI’s ArcGIS Web and Mobile SDK’s, this includes the Flex (Flash builder 4), Silverlight API’s, Java Script API. This architecture also includes Latitude Geographics’ Geocortex Essentials layered over the ESRI rest based services. ArcGIS Server 10.1 anticipated early 2nd quarter of 2013.

**Primary Architecture for services tier:** Microsoft Windows 2008 64 Server & .NET 4.0 and above.

**Application development language:** C# (MS Visual Studio 2012, 2010, and 2008 for MS 6 Mobile)

**Scripting Language:** JavaScript (jQuery), Python, MS Power shell.

**Reports solution:** SAP Crystal Reports for Visual Studio, or Oracles OBIEE

Note that Oracle OBIEE is provided by a separate data delivery group at MNIT Services @ DOT.

**Database:** Oracle 11g2

**Database Middleware:** ArcSDE (at 10sp5, **and not 10.1 in 2nd quarter 2013**) with various

Geometry Direct Connects. DevArt dotConnect for Oracle and DevArt OraDevelopers tools for Visual Studio 2010 are used for those needed Microsoft entity framework tools. Oracles ODP.NET for entity framework are tested at MnDOT now that they are released to production. ODP.NET for entity framework can be used. ODP.NET is preferred.
Object/Relational mapping: ESRI’s ArcSDE, ESRI’s ArcObjects, and .NET, Microsoft’s Entity Framework. (see above for versions).

Controller framework: MS .NET 4.0 (et al), ESRI’s Web and Mobile .NET with its supported API’s. Latitudes Geographics’ Geocortex Essentials are also available.

View / presentation framework: Windows XP or Windows 7 PC with .NET 4.0
Windows Internet Explorer 8 and Windows IIS Server
Windows Mobile Device with Compact Framework 2.0
HTML, CSS, and ASP.NET Technologies. Note that HTML5 (IE 9.0, Chrome), Android OS, and iOS are starting to emerge within MnDOT.

ASP.NET application server: MS .Net 4.0
Code version control system / repository: MnDOT’s CVS
Detail modeling / design tool: SAP’s Power Designer is now a part of NEW project Development. All database designs for new work must go through an initial approval process by MN.IT @DOT Data Delivery Staff. Data requirements are delivered in a separated document. Data, its design, and approval are SIGNIFICANT tasks at MnDOT. Please plan for it proactively.

Primary testing tools: MS Visual Studio 2010, 2012 Pro Ed. tools, Premium Edition Tools are available for code reviews. OpNet and Solar Winds application monitoring technologies are also available for comprehensive enterprise environment testing.

CVS interface tools: Tortoise CVS.
Defect / Issue tracking tool: Atlassian JIRA (requires AD account)

Terminology used above is based on Model, View, Controller Architectural Pattern used in MN.IT @DOT Enterprise Services section for web based applications. Using Microsoft’s ASP.NET MVC framework is a welcome practice. Please ask for a template web application for web based work.

Enterprise Architecture – Logical Services Diagram for both .Net and Java

The primary focus of the GIS development framework will be .NET based. The reality on the ground is that MnDOT’s current Enterprise Architecture is and will be for the foreseeable future, a mixture of Java/J2EE and Microsoft .NET components, as well as emerging mobile technologies. Projects, with needs for Web and Mobile technologies, can take some direct advantage of this mixture. The Enterprise Architecture is presented here to give a picture of what the .NET and JAVA environment looks like. There may be some interoperability offered by Web Services and common SOAP/REST and XML protocols in both architectures, this is extremely powerful and leverages our existing resources to benefit our customers. Web Service supported development or centric development is welcome.
Development / Maintenance Environment Model:

The following are required learning for those new to developing software at MnDOT. Obtaining proper access to systems is required to build, deploy and maintain a web or mobile solution using ArcGIS Server at MnDOT:

- Developer must be aware of MnDOT and MN.IT policies as concerns data, security, and the dev, test, production progression of software development.
- Developer must have an AD (active directory) account (all employees have this, all contractors must get this via an access authority).
- Developers must be IT staff or contracted IT staff.
- Obtain Application Server access via MN.IT @ DOT Enterprise Services staff.
- Obtain enterprise spatial database access via MN.IT @ DOT Enterprise Services staff.
- Obtain issue tracking to JIRA by having AD account placed in proper group. Via MN.IT @ DOT Enterprise Services staff.
- Obtain a development desktop/laptop that has ArcGIS 10sp5 as well as development tools as necessary to work with the ArcGIS servers and Oracle databases.
• Understand ArcGIS web based services and how to author and provision.

• Understand how development work progresses from developer credentials to project credentials (MN.IT @ DOT Enterprise Services staff will assist), at the development level.

• Understand how development work progresses from project credentials at the development level to the test level.

• Provide management tools for data and application management that allow for appending to application management data (no schema changes once project credentials are used).

• Provide software and written processes to construct/deploy data structures, to install developed application code, to configure database and applications, to configure the application platform (IIS application pools, security, network proxies), and to seed or load sufficient data as agreed to allow testing of the application.

4. GIS Application Standards
MN.IT @ DOT Enterprise Services has specifications for GIS Applications that software developers will follow. These include:

Coding Standards:
Web and Mobile GIS Developers will follow coding practices that provide the following:

• Easy to understand and read source code.
• Clear and concise logical and physical system models
• Clear and concise logical/conceptual and physical data models
• Use the CVS for all works, and be sure code can be extracted, built, configured, and deployed from the CVS source and all stored there.
• Frameworks and Patterns to assist in application structure understanding and interfaces.

Graphics User Interface (GUI) Standards:
GUI standards follow Microsoft .NET GUI recommendations in combination with ESRI’s standard industry practices as they relate to GIS and various devices as a base.

From the design process, the MnDOT Office of Communications provides a document on External reaching Web Site Rules and Goals that covers:

Structure
Tools
Rules for Content, Writing and Style
Logos, Fonts, Colors, Meta Tagging, Tables.
Video, PDFS, Audio, Power points
Applications items, FTP locations for large transfers
Naming and Hosting.
Americans with Disabilities Act Accessibility
And all items web site construction related.
MnDOT's Web Template Application includes pages that meet GUI standards. It's best to start with these pages and a thorough review of the Office of Communications External Guidelines as mentioned above. Note that these apply to the initial or splash page for external sites. See your project contact for these materials. Applications are not specifically held to these standards.

For desktop applications, the design process will introduce required specifications in the areas of:

- Screen Resolution
- Screen Layout
- Navigation Bars
- Logos Usage
- Security (Pre-Login, Login, Logout)
- Fonts and Colors
- Graphics
- Error and Messaging
- All Controls for Application (GIS and standard)
- Windows and panels.

**Versioning Standards:**

Developers will follow Apache Software Foundation’s versioning guidelines. Located at [http://apr.apache.org/versioning.html](http://apr.apache.org/versioning.html)

OI&TS's simple description of versioning guidelines are as follows:

A `<Major>-<Minor>-<Patch>` numbering system is recommended to track software revisions when a new version is released. Note that CVS system has its own internal revision numbering system, and it will not match this standard. Use CVS Tags to assign this `<Major>-<Minor>-<Patch>` numbering system within the CVS storage system.

A Major Release is a full product upgrade of the software containing significant new functionality. A Major Release is necessary if there are changes to the model or compatibility with previous versions cannot be maintained. When the Major Release version number is incremented, the Minor and Patch version numbers are reset to zero.

A Minor Release is a planned update to the existing software incorporating standard maintenance, improvements to existing features, enhancements and bug fixes. When a Minor Release number is incremented, the Major Release version number remains unchanged and the Patch Release version number is reset to zero.

A Patch Release is distributed when necessary to correct critical or significant problems that impact a customer’s use of the system. When the Patch Release version number is incremented, the Major and Minor Release version numbers remain unchanged.
Software build, dependency, and version management:

- Developers will use Visual Studio 2008, 2010 SP1, or VS2012 and CVS for build, dependency, and version management when building Mobile and Custom works for Web Based applications.
- Developers will build .MSI or EXE files or appropriate scripts for deployment operations of the clients and associated data for the services and the ASP.NET web site structures for the web site files.
- Developers will at all times consider the automation of deployments to speed the process of deploying and keeping this critical process moving as fast as possible.

Source Code Maintenance:

Developers will provide and maintain application source files, configuration files, database schemas and scripts, developer documentation, and other related artifacts in MnDOT’s CVS.

Follow these rules for naming CVS Modules:

One module (directory) per application (including all variants of that application)

The top level module name should be short (3-8 character if possible), all lower case letters and/or digits with no "-"(dash), and "_"(underscore), "."(period), or spaces. The module name may also be used in the JIRA project key, database schema name, and/or AD role name

The module name is short to facilitate naming of resources and artifacts where long user-friendly names would be problematic. For example, database table names are often pre-pended with the application name. Projects should determine the application's name (typically an acronym) early in the project so project artifacts and configuration files can use this name.

Use of CVS Tagging will allow for the Major Minor Patch pattern. The auto revision number that CVS assigns will be different than the Major Minor Patch pattern and should be left alone.

Documentation Tags (Visual Studio XML Document Tags):

It is suggested that developers will use XML Document tags or suitable source code comment notation for automatic assembly of documentation from source codes in C# with automated tools.

Object / Relational Mapping (SDE and ADO.Net):

Developers will use ADO.Net and ESRI’s SDE technologies for Object / Relational Mapping for tabular and geographic data. Oracles SDO geometries are default in the SDE environment, likewise, SDE Binary and ST geometries can also be used, as they provide certain benefits depending on what the work at hand is. DevArt tools for Microsoft Entity Framework usage are licensed at MnDOT for enterprise usage. Oracle ODP.NET now has drivers that allow for Microsoft Entity Framework usage and are allowed, these give the same functionality as DevArt tools in this area, for no extra cost.
Code Portability:
Developers will follow practices that insure code portability. Peer reviews and documented justification is required for the use of proprietary application server extensions or libraries. If such libraries are used, it’s suggested to use NuGET to keep these up to date via VS2010, VS 2012 extensions. Developers MUST ALWAYS collaborate with MN.IT @ DOT Enterprise Services staff on technology libraries they wish to introduce. MN.IT @ DOT Enterprise Services decides what will be used.

Deployment:
Please consult with the Project Manager and MN.IT @ DOT Enterprise Services staff for doing an initial deployment from Developer credentials to Project credentials. Good and fast deployments take planning and coordination amongst a number of people and groups.

Auditing:
Models and code will be reviewed for compliance to the specifications listed in Section 3.

Controller Standards
Developers will use MS .NET 4.0, ESRI’s Web .NET, ESRI’s Mobile .NET, and Latitude Geographic’s Geocortex® Essentials framework for most web and mobile applications work.

Application Security:
Developers will use the appropriate security mechanisms as per project needs. There is a wide range of tools and techniques and many considerations. Security will be a primary consideration and requirement for all projects “upfront” and not a last minute consideration. All levels of security are to be considered for all applications.

Items to consider:
- Separate Administration application from user application
- Awareness of OWASP top ten web based security issues, and active consideration to avoid these issues.

Scripting Standards
Developers will use JavaScript for client-side scripting, and JavaScript or Python for server-side scripting. jQuery and JavaScript libraries friendly to ESRI development are to be used.

Naming Standards for Web applications (‘User friendly’ name)
The user-friendly name is the descriptive or popular name that identifies the application. The user-friendly name should be chosen by the project sponsor, application owner, with the project manager. Projects should determine the application's user-friendly name early in the project so it can be used in documentation.

URL
The URL is the application's Web address. Projects should determine the URL for the application. The URL is typically based on the user-friendly name, and should follow these examples:

- Production URL = appname.state.mn.us
- Test URL = appnamet.state.mn.us
- Development URL = appnamed.state.mn.us

**Short-name**

The application short-name is used to name application and data resources and artifacts (CVS) where long user-friendly names would be problematic. For example, database schema names are often pre-pended with the application short-name.

Projects should determine the application's short-name (typically an acronym) early in the project so project artifacts and configuration files can use this name.

**Oracle Data Storage and Deployment Progression**

Applications data will always start out on a development schema on a development server offered by a MN.IT @ DOT Enterprise Services unit (and arranged by staff). This data will then progress to a test and finally to a production version, losing the ability to alter schema (make any data definition language changes) while in “Test” or “Production”.

Good solid data normalization is required to allow for expansion of the application without altering the schema. Typically some kind of application data management tool is needed as well, or at least a clear manual process needs description. Furthermore, meta data in 2 distinct forms is required, one set is in Oracle Meta data and Comments and one set in ESRI’s formats suitable for inclusion to a Metadata portal application.

A data base standards document will be provided. This document deals with commenting and proper object naming.

There are important items to consider when existing MnDOT applications are being outfitted with GIS functionality, and these may or may not be in the Scope of Work (SOW) that has been developed or in the Project Plan the work is being performed under.

Ideally, GIS (or other integrations) will have been considered in the development of the original application, however, this is not always the case and many existing applications have their data in non-spatial capable instances of Oracle. Specifically, pay attention to the following items and be prepared to ask requirements types of questions for:

- Movement of existing application data to Oracle instances that support spatial data are desired and will almost always need to occur (TGP instances).
- Existing Data Integrity is a big issue. Having more than one application able to update the same data introduces many problem areas. Good business process integrity (how data is edited (insert, updates, deletes)) and by who can prevent collisions, but these processes tend to change over time and a false sense of good data integrity might occur if the
software isn’t designed to check for these business rules related to collisions. The following designs guides should be considered always:

- Always use the existing applications API to update/add data, when they are provided.
- Shorten the transaction timings. GIS and specifically MOBILE GIS has the concept of long transactions (those lasting many minutes to days).
- GIS data alone can use ESRI’s VERSIONING (and attendant Reconcile and Post) operations. Existing tabular data under business rules probably cannot take advantage of this mechanism.
- Consider using LOCKS on any existing business data that goes out with GIS and MOBILE GIS edit sessions, so that NO ONE can update until that data comes back, and locks are released. Consider the Management tools needed to unlock records and track editors and their checkouts.
- Consider mechanisms to perform DELTA change detection, along the lines of staging tables, so that changes can be seen.
- Consider using “additive inserts only” data design and application data flow for editing. This is an asset/sample or inspection type of scenario. In order to get the most current data about that asset or sample/inspection taken, a MAX function on DATE is needed. All inspections are available with this concept of data design, all are stored.
- Consider using “view” only “Mapit now” buttons in the application to tie the GIS and existing business application together. Geography and business attributes are loosely joined and each is edited in separate processes. The inability to join data will point out where bad data is or where the two editing processes are out of sync.

- Database account patterns will be along the following examples table. All ArcGIS services get a separate account. These patterns exist to help see who or what is working against the various data. This is not the full list, but a good representative example.

<p>| AGS_????? | ArcGIS Service account, used by ArcGIS server to access oracle. General Map services |
| AGS_?????<em>MOBILE | ArcGIS Service account, used to ArcGIS server to access oracle for MOBILE services. |
| AGS</em>?????_!!!!!! | ArcGIS Service Account, used special access as described by !!!!! |
| MCCA1CHA | Developers Personal account on TGD, this one for Charlie McCarty (as represented in the AD domain). |
| ????_ADMIN | All schema controlled by this account – for DBA’s to control all other project accounts |</p>
<table>
<thead>
<tr>
<th>Account Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>?????_USER</td>
<td>CRUD or as needed access by this account – for non ArcGIS oracle access.</td>
</tr>
<tr>
<td>?????-BUILD_USER</td>
<td>CRUD or as needed access by this account, for iterations where the new iteration of development gets done in the “BUILD” version and then deployed to the non-build account, after that development has been moved up to the higher TEST level.</td>
</tr>
<tr>
<td>?????_MAINT_USER</td>
<td>CRUD Maintenance account, used to maintain tables, views, lookup tables, by the maintenance tools when in test and production.</td>
</tr>
<tr>
<td>?????_LOAD_USER</td>
<td>Used for applications that must load data in bulk to tables/objects for further processing on the Oracle side.</td>
</tr>
<tr>
<td>?????_REPORTS</td>
<td>Used for report writing</td>
</tr>
</tbody>
</table>

- MAINT accounts in Oracle can be set up for ArcCatalog specifically to deal with DOMAINS and SUBTYPES – it owns the GB Domain works, and can be established for just the purpose of keeping domains up to date. This account updates the DOMAIN values as needed, at TEST and PROD levels, using a tool like ArcCatalog if necessary, or a Management page. All other accounts simply use its data. The Project acronym should be used in prefix of name of all ArcGIS domains so we know what they are for, when looking at the bunch in ArcCatalog or other tools.

**Issue / Bug Tracking (JIRA)**
Developers will use [JIRA](http://jira) for software issue / software bug tracking.
MnDOT’s JIRA is located at [http://jira](http://jira). You must be connected to a MnDOT network for access.

**Developer Team Collaboration (Wiki)**
MN.IT @ DOT Enterprise Services has a [Wiki](http://wiki.dot.state.mn.us) available for collaboration, documentation, and real-time information exchange. Be aware this Wiki is not a production service. Availability is not guaranteed.

MN.IT @ DOT Enterprise Services Wiki is located at [http://wiki.dot.state.mn.us](http://wiki.dot.state.mn.us). You must be connected to a MnDOT network for access.

It is hoped that MS SharePoint and Team Foundation Server abilities will soon be available for project collaboration.

**Logging Standards**
Developers should use Standard Windows Logging methods for capture of application and installation events of interest. For web based applications, a special directory is created for logs and messages is created called g:\arcgisserver\arcgiserror, and in here log files of application
interest can be created with the project name abbreviation, or functional units abbreviation, to keep them separate from other projects.

MN.IT @ DOT Enterprise Services Network Standards:
MnDOT has a network policy of least privilege when it comes to access of resources by people, processes, and devices. This means that the FEWEST privileges will be assigned that are consistent with the duties and functions of the software developed. Protocols not needed, messaging formats not needed, will not be available to use. SOAP is recognized, but is pushing the edge of what is allowable as for rich messaging. There are many items to consider in this area, including reverse proxies usage for external facing apps, ftp, email, and ssh, as well specific software packages. For SMTP email to work from a web based application, the server needs to have McAffee Virus Scan Access Protection Rules modified to allow this to happen, in addition to other asp.net and code setups.

All web based applications for MnDOT must be aware of network proxies to access resources correctly. For ASP.NET applications, this means the following type tags are needed in the web.config file:

```xml
<System.Net>
  <defaultProxy>
    <proxy autoDetect="True"/>
  </defaultProxy>
</System.Net>
```

5. Hosting Environment:
Application hosting services are typically provided by MN.IT @ DOT Enterprise Services Infrastructure Section. MN.IT @ DOT Enterprise Services Applications are hosted by Windows IIS and ArcGIS Server on Windows 2008 R2 64 bit servers.

Database hosting services are typically provided by MN.IT @ DOT Enterprise Services Infrastructure Section. Databases reside on a storage area network (SAN) managed by an Oracle RDBMS running on Linux as RAC server.

Reports hosting services are typically provided by OI&TS's Infrastructure Section. The reporting solution is typically Crystal Enterprise.

All servers are likely to be virtual, and will be established in one of two data centers connected to SAN storage and very high speed networks.

6. Development / Test Environment
Unit testing is done on development servers. Integration testing is done on development servers. System testing and user acceptance testing is done on test servers that reside in the same environment as the development servers. Developers and business unit staff are responsible for all testing. Code must reside in MnDOT’s CVS repository and compile at MnDOT in MnDOTS environment, and perform according to specifications.

7. Production Environment

Production instances are deployed on production servers that typically reside in the same environment as the development and test servers.

8. Security

When external user access is required.

An external user means a non MnDOT employee who from time to time use a specific internal MnDOT application. Often external users are other government workers, contracted staff, or professional staff (engineers) who have some kind of direct business with MnDOT.

MN.IT @ DOT Enterprise Services has SOME directory service (AD) for external users (at this time). An external user is NOT a MnDOT employee.

External users will be either automatically allowed to access web systems as an anonymous user, or will need to exist in an application database.

If an external user authentication and authorization will be implemented, user credentials and roles will reside in the application database, using the concepts of role based security. This is accomplished, for example, by having Oracle tables accessible from the application that store all pertinent information about the users and the roles they carry. When a user logs in, their credentials are verified against information in these Oracle tables and other user persisted data is accessed and used. This table or list is then maintained by a set of administrators that have administrative level roles on the system. These administrators can add, delete, update, and approve users for access. The Oracle tables can be used to support self-help routines for regular users with none or existing credentials, to email a forgotten or new password, or to request a user account. Some administrative tool coding will be required.

Credentials used for authentication and authorization must be encrypted, when used from outside the MnDOT network. Using HTTPS is one way to ensure this, ASP.NET membership services and configurations have another way. External browser to server communication may be encrypted. MnDOT will use 128 bit encryption via SSL.

When Secure user access is required from outside the MnDOT network.
Again, credentials used for authentication and authorization must be encrypted, when used from outside
the MnDOT network. In this case, the AD domain can be used as the source of data to compare to, and
using ASP.NET membership services is the recommended way forward.

**MnDOT web application development security guidance.**

MnDOT web application development will be developed with OWASP top ten web security issues in
mind and all efforts to address the OWASP top ten web security issues will be taken into consideration as
software is developed. The OWASP top ten are:

1) Cross Site Scripting (XSS)
2) Injection Flaws (of all types, ie SQL, LDAP, XPath, XSLT, XML, OS Command)
3) Malicious File Execution
4) Insecure Direct Object Reference
5) Cross Site Request Forgery (CSRF)
6) Information Leakage and Error Handling
7) Broken Authentication and Session Management
8) Insecure Cryptographic Storage
9) Insecure Communications
10) Failure to Restrict URL access

Understand what they are and how to avoid.

9. MnDOT Personal computer and mobile field units.

A wide variety of Personal Computer and Mobile Field Collection units are capable of performing work
that conforms to practices with in MnDOT. All hardware considerations for all projects should be
reviewed by the project manager and system architect as part of the projects scope.

For field units, with a GPS involved, it is desirable to be able to achieve 1 meter or better location
accuracy, which means some form of reliable differential correction (real time or post process) must be
used. WAAS, Beacon, or VRS are typical real-time correction sources.

Recently, many new tablet devices (Apple iPAD, Android based) and mobile phone devices have gained
usage in MnDOT via exceptions. A generic Mobile framework is in development. If you are doing any
mobile GIS/GPS work, please check in with MN.IT @ DOT Enterprise Services staff on the progress of
these developments, as you will likely need to consider them in your works.