FDOT Development Environment
Effective 8/30/2017

1. Technology Requirements

1.1 All source code must be compatible with the most current retail release of Visual Studio Professional or its preceding version. For instance, if Visual Studio 2017 is the most recent retail release, then the source must be compatible with at least Visual Studio 2015.

1.2 All source code will be stored in a Git repository hosted in FDOT's Visual Studio Team Services (VSTS) account.

- **1.2.1** It is recommended that the latest code is committed to source control at least once a week.
- **1.2.2** FDOT retains the right to request a full code commit at any time.
- **1.2.3** A tagged commit must be done in coordination with all Unit Test publishes.
- **1.2.4** All System Test publishes will be done with builds from tagged versions from the code repository.

1.3 All web applications/websites must be designed to primarily be hosted in Microsoft Azure's PaaS offerings.

- **1.3.1** Any parts of the system that are strongly dependent on on premise infrastructure should be separated from the rest of the system so they can be deployed on premise and integrate with the larger whole through a service bus. All on premise deployments should be deployable on Windows Server 2016.

1.4 All code must conform to Microsoft's C# Coding Conventions and must follow Microsoft's .NET Framework Guidelines and Best Practices wherever possible.

1.5 All code must conform to the current FDOT .NET Code Review Standards.

2. Publishing

2.1 Build and release of solutions must be handled by VSTS's Build and Release platform.

- **2.1.1** Builds will be the responsibility of the vendor (outsource contracts) or programming team (internally staffed projects).
- **2.1.2** Releases will be a shared responsibility of both the vendor, programming team, and FDOT server and architecture staff.
- **2.1.3** FDOT staff will be made available to assist in the build and release process.

2.2 The development team will have the authority to publish their code to the project's Unit Test website.

2.3 System Test publishes will be performed by a FDOT Application Architect.

2.4 Production deployments will be handled by designated FDOT staff and based on the last version deployed to system test. This will generally be a new build from the same tagged release that was deployed to system test.

- **2.4.1** Approvals by the Functional Application Coordinator, FDOT Team Manager, and FDOT Team Server Administrator will be required to release the Production Build.

2.5 Batch Job publishes in both Unit Test and System Test will be performed by the FDOT Architect from a specific Subversion code version in an approved FDOT code repository.

2.6 Batch Job publishes to Production will be performed by SSRC staff via the FDOT Architect, and will be from current System Test code.

3. Infrastructure

3.1 **Unit Test** – First level testing environment. This is where developers can perform integration testing to ensure that the application works correctly in FDOT's server environment and with the other enterprise systems it might be interacting with. All databases typically have an instance at this level to facilitate testing. These databases are typically refreshed as needed from System Test or Production.

3.2 **System Test** – Second level testing environment. This is where the Users will perform their User Acceptance testing and the staging environment for Production migrations. All databases typically have

an instance at this level to facilitate testing. These databases are typically refreshed monthly from Production (for all applications that are live in Production).

3.3 **Production** – Live environment.

3.4 All new web servers must be Windows Server 2012 R2 or 2016.

3.5 FDOT uses multiple Active Directory domains that all reside in a single Forest w/ Trust

3.5.1 An FDOT Architect can provide more info related to this if needed.

4. FDOT Enterprise Library (FEL)

FDOT has a code library that can be utilized by any application for various purposes. Some parts of the library are required to be used by all applications for various functions. Other parts are required to be used if interactions with certain Enterprise systems are required. Additional parts of the library are for some common functionality, but are not required to be used. There are three versions of the FEL Library that have been developed. Detailed documentation on all versions of the FEL is available at http://interdev.dot.state.fl.us/wiki/. Unless otherwise noted or required for a given project, the most recent version of all enterprise services must be used where supported.

FEL 2: FEL 2 is the original version of the FDOT Enterprise Library. FEL 2 is still used in many Enterprise Applications. FEL 2 is **not** service-based. FEL 2 makes all calls directly to the enterprise databases, and therefore is **not** usable outside of the FDOT firewall.

FEL 3: FEL 3 started the move to service-based delivery. Expansion of FEL 3 eventually became FEL 4. For this reason, FEL 3 has fewer components and is not found in many Enterprise Applications. FEL 3 is also **not** usable outside of the FDOT firewall.

FEL 4: FEL 4 is the most current version of the FDOT Enterprise Library, and is the version required for all new development. FEL 4 is service based. Test implementations of most of the FEL 4 services are available over the internet to facilitate development when Vendors are coding and not connected to the FDOT internal network.

4.1 **Required enterprise services:** Developers are **required** to use the following enterprise services:

4.1.1 **Connection Strings** – all connection strings must be obtained from the connection string provider by connection label. Vendors will need to work with the FDOT DBAT assigned to their project to obtain connection labels to be used by their project. This same approach also applies to user id/password combinations if an application needs an account for some process (i.e. connecting to a 3rd party web service).

4.1.2 **Error handling** – custom error handling is applied at the server to all web applications and handles unhandled server side exceptions by logging and sending emails to the configured email groups. Applications must provide the necessary configuration settings for this to work properly.

4.1.3 **Email** – any emails being sent from the application must obtain SMTP settings from the FEL configuration provider so changes across the enterprise do not affect applications individually.

4.1.4 **FDOT Security Token Service (STS)** - builds, signs, and issues security tokens according to the WS-Trust and WS-Federation protocols. The FDOT STS supports all authentication methods approved for FDOT applications. This includes Active Directory and Internet Subscriber Accounts. Unless otherwise specified, the STS is required for all authentication.

4.1.4.1 Applications deployed on the FDOT Intranet using Active Directory only are not required to use the STS, as Windows Authentication is an acceptable alternative.

4.1.4.2 Enterprise services for authentication. When the STS requirement is explicitly dropped for a project, the following services must be used:

4.1.4.2.1 Internet Subscriber Accounts (ISA) – mechanism for non-FDOT users to create and maintain an account to access various FDOT systems.

4.1.4.2.2 RACF – an internal authentication/authorization system.

4.2 **Required-as-needed enterprise services:** Developers will be required to use the following FEL services for certain system interactions. Note that for a given project, some of these services may not

be necessary, but where the information and functionality they provide are needed, these services must be used.

4.2.1 Staff Repository System (SRS) – the repository of all internal and external staff.

4.2.2 DOT Code Tables (DOTCODES) – access to Enterprise Code Tables.

4.2.3 Enterprise Document Management System (EDMS) – storage mechanism for all electronic documents.

4.2.4 Organization Codes (OrgCodes) – database of FDOT's organizational units.

4.2.5 Financial Project – database of FDOT's construction projects.

4.2.6 Active Directory – service interface for Active Directory information (e.g. group lookups, user information, etc.).

4.3 FEL 4 Optional Components: Use of the following FEL 4 components are optional:

4.3.1 **Helper Functions** – there are a handful of functions that are not required but can be useful. One of the most useful is the EnvironmentLevel property. This returns which level the server is that the application is running on and can be mainly used to run code for the test environments that should not run in Production. See documentation for a complete list.

4.3.2 **Diagnostics** – A package with code-free support for routing System.Diagnostics.Trace to Log4net and producing SQL output from NHibernate for SQL reviews. See documentation for details.

5. Approved Technologies

5.1 Anything not on this list must be approved for use on a per-project basis.

5.2 Use of approved technologies does not exempt the project from meeting all other FDOT standards.

5.3 Vendors may not use third-party tools or code which would place a licensing responsibility on FDOT without prior review and approval of the tools or code by OIT. Unless otherwise specified, the vendor is responsible for providing any licenses required for deployment and at least one development license for use by the FDOT application maintenance team.

5.4 Any 3rd party libraries must be included via NuGet where possible. If not possible, then the DLLs must be included with the source.

5.5 Languages

5.5.1 C# 6.0+ (64-bit is required; 32-bit solutions require an exception)

5.5.2 JavaScript

5.6 Project Types

5.6.1 .NET or .NET Core

5.6.2 MVC 5.0+ w/ Razor Syntax

5.6.2.1 This includes everything that is typically included by Microsoft in an MVC project.

5.6.3 Web API

5.6.4 Console application (.exe) for any Batch Jobs.

5.6.5 Class Library

5.6.6 Test Projects

5.6.6.1 Microsoft Unit and Coded UI Test Frameworks

5.6.6.2 NUnit

5.7 Libraries

5.7.1 All core .NET Framework Libraries included in the standard .NET or .NET Core install.

5.7.2 NHibernate or Entity Framework

5.7.2.1 Two ORMs are not allowed in the same project.

5.7.3 ADO.Net with parameterized queries if not using an ORM

5.7.3.1 Stored procedures are not allowed.

5.7.4 Oracle DataAccess Driver (Oracle.DataAccess.dll - version matching the version FDOT is currently using) must be used for all Oracle database access.

5.7.5 IBM DB2Connect Driver (version matching the version FDOT is currently using) must be used for all DB2 database access.
5.7.6 Ninject – dependency injection.
5.7.7 Automapper – object to object mapping.
5.7.8 Glimpse – web site debugger.
5.7.9 Moq – a testing framework for mocking dependencies.
5.7.10 Open XML SDK – Word, Excel, and PowerPoint exports.
5.7.11 Closed XML SDK – Excel exports
5.7.12 PDF Generation
5.7.12.1 iText#
5.7.12.2 Muhimbi PDF Converter Services (document conversion, merging, and watermarking).
5.7.13 Log4Net – Logging framework.
5.7.14 Newtonsoft JSON.Net – JSON serialization library.
5.7.15 Sharp Serializer – Efficient multi-format .Net serialization library.
5.7.16 CSS Frameworks.

5.7.16.1 Bootstrap
5.7.17 jQuery and any jQuery plug-ins.
5.7.18 All non-minified source must be provided.
5.7.19 Any other 3rd party JavaScript libraries are also acceptable, provided that all source is included and license requirements are reviewed and accepted by FDOT (an IRR must be submitted and approved by FDOT through the FDOT Contract Manager).
5.7.20 Microsoft SQL Server Reporting Services Client for any reports.
5.7.21 Telerik UI for ASP.NET MVC.
5.7.22 AngularJS v1.6.X
5.7.23 Angular v4.X

6. Exceptions

6.1 All requested exceptions to these standards shall be submitted to the Technology Standards Review Team (TSRT) in writing. The TSRT will review each exception request and send a recommendation to the Enterprise Architect, who will make the final decision.