Deltares

DAM

Architecture Overall

DAM
Architecture Overall

Authors
Tom The

Partners
Irene van der Zwan

DAM

Architecture Overall

Client	Deltares - Geo engineering DKS
Contact	Irene van der Zwan
Reference	1210702-000-GEO-0005
Keywords	Dike, safety assessment, design, software, macro stability, piping

Document control	
Version	1.0
Date	Jul. 2019
Project number	1210702-000
Document ID	Repos\dam\DamOverall\trunk\doc\DAM - Architecture Overall\DAM - Architecture Overall.pdf
Pages	24
Status	final

Author(s)		
	Tom The	Deltares

Doc. version	Author	Reviewer	Approver	Publish
1.0	Tom The	John Bokma	Maya Sule	-

Executive summary

This document contains a description of the overall architecture for DAM, an application that computes the strength of a complete dikering with respect to several failure mechanisms, such as macro stability and piping.

Samenvatting

Dit document bevat een beschrijving van de totale architectuur van DAM, een User Interface applicatie die een gebruiker in staat stelt om voor een dijktraject berekeningen uit te voeren voor verschillende faalmechanismen, waaronder macrostabiliteit en piping.

About Deltares

Deltares is an independent institute for applied research in the field of water and subsurface. Throughout the world, we work on smart solutions, innovations and applications for people, environment and society. Our main focus is on deltas, coastal regions and river basins. Managing these densely populated and vulnerable areas is complex, which is why we work closely with governments, businesses, other research institutes and universities at home and abroad. Our motto is 'Enabling Delta Life'.

As an applied research institute, the success of Deltares can be measured in the extent to which our expert knowledge can be used in and for society. As Deltares we aim at excellence in our expertise and advice, where we always take the consequences of our results for environment and society into consideration.

All contracts and projects contribute to the consolidation of our knowledge base. We look from a long-term perspective at contributions to the solutions for now. We believe in openness and transparency, as is evident from the free availability of our software and models. Open source works, is our firm conviction.

Deltares is based in Delft and Utrecht, the Netherlands. We employ over 800 people who represent some 40 nationalities. We have branch and project offices in Australia, Indonesia, New Zealand, the Philippines, Singapore, the United Arab Emirates and Vietnam. In the USA Deltares also has an affiliated organization.

www.deltares.nl

Contents

List of Figures

List of Tables

1 Introduction

1.1 Purpose and scope of this document

This document contains the overall architecture of DAM, a software package for the automated calculation of the strength of dikes, and all of its components. DAM was developed by Deltares with and for STOWA for all water authorities.

1.2 Other system documents

The full documentation on the program comprises the following documents.

Title	Content
DAM- Architecture Overall (**?**)	Description of overall architecture of DAM and its components.
DAM Engine - Technical Design (**?**)	Description of the implementation of the architecture and technical design of the DAM Engine.
DAM Engine - Technical documentation (**?**)	Description of the arguments and usage of different software components of the DAM Engine, generated from in-line comment with Doxygen.
DAM Engine - Test Plan (**?**)	Description of the different regression and acceptation tests of the DAM Engine, including target values.
DAM Engine - Test Report (**?**)	Description of the test results (benchmarks and test scripts) of the DAM Engine.
DAM UI - Functional Design (**?**)	Description of the requirements and functional design of the DAM User Interface.
DAM UI - Technical Design (**?**)	Description of the implementation of the architecture and technical design of the DAM User Interface.
DAM UI - Technical documentation (**?**)	Description of the arguments and usage of different software components of the DAM User Interface, generated from in-line comment with Doxygen.
DAM UI - Test Plan (**?**)	Description of the different regression and acceptation tests, including target values for the DAM User Interface.
DAM UI - Test Report (**?**)	Description of the test results (benchmarks and test scripts) of the DAM User Interface.
DAM UI - User Manual (**?**)	Description of the different functionalites available in the *User Interface* and background information.

Table 1 DAM system documents.

2 Technical requirements

2.1 Platform

Operating system: Windows 7- 32 bits or Windows 7- 64 bits, Dutch or UK version.
Read/write permission on user selectable folder for saving and calculating projects.
Administrator rights are required for installation of the application.
Required: Microsoft DotNet framework, version 4.5.

2.2 System requirements

Processor: Intel Core i5 or better.
Clockspeed processor: 2.4 GHz or better.
Memory (RAM): 4 GB or more.
Free harddisk space: 20 GB or more.
Monitor: 22 inch monitor, resolution 1920x1080.

2.3 Regional settings

The user can select between 2 languages in the application:

- Dutch (NL)
- English (UK)

The application should behave correctly with regional settings set to:

- UK
- NL

2.4 Additional requirements

- DAM must support the use of multiple processor cores, when they are available, to speed up the calculation.
- DAM is a standalone application, but must support the use of network storage.

3 DAM and its components

DAM contains several components. Please see Figure 1 for an overview of the components.

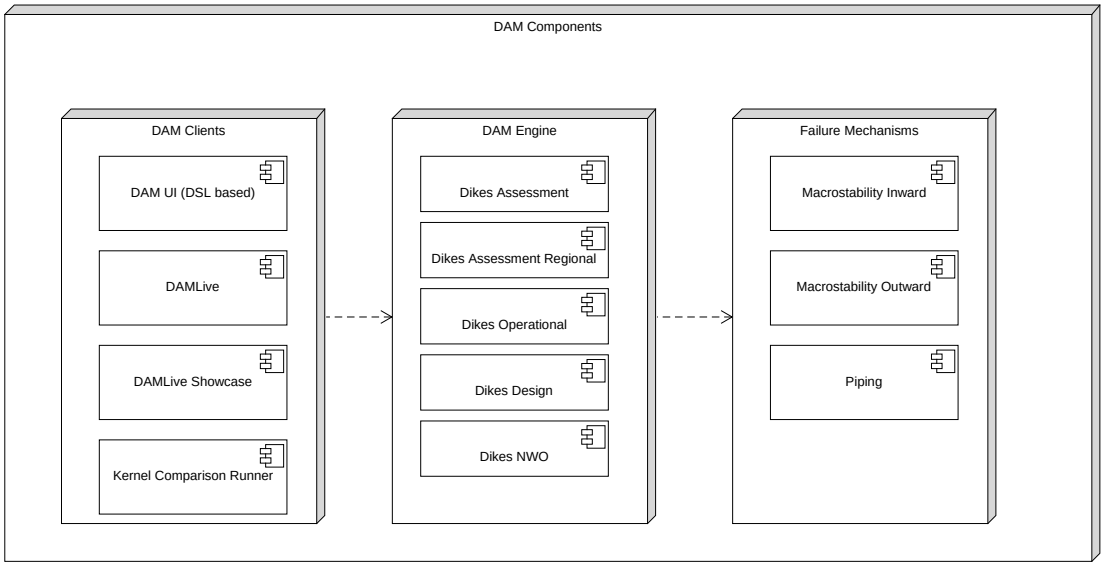

Figure 1 DAM and its components.

The arrows illustrate the dependencies of the components. A dependency also implies communication and interaction between the respective components. The exact definition of how to communicate and interact between these components is described in the technical design of each component (e.g for the DAM Engine (**?**)).
In the following sections the components are described.

3.1 DAM clients

DAM clients are the modules that mostly interact with the user or sometimes with another system. These can be full graphical user interfaces (like DAM UI as shown in Figure 2 and DamLive Showcase as shown in Figure 3 and Figure 4), commandline parameter tools (like KernelComparisonRunner.exe) or a module that can be used by other systems (like DAMLive, that is to be used as a module in a FEWS system).

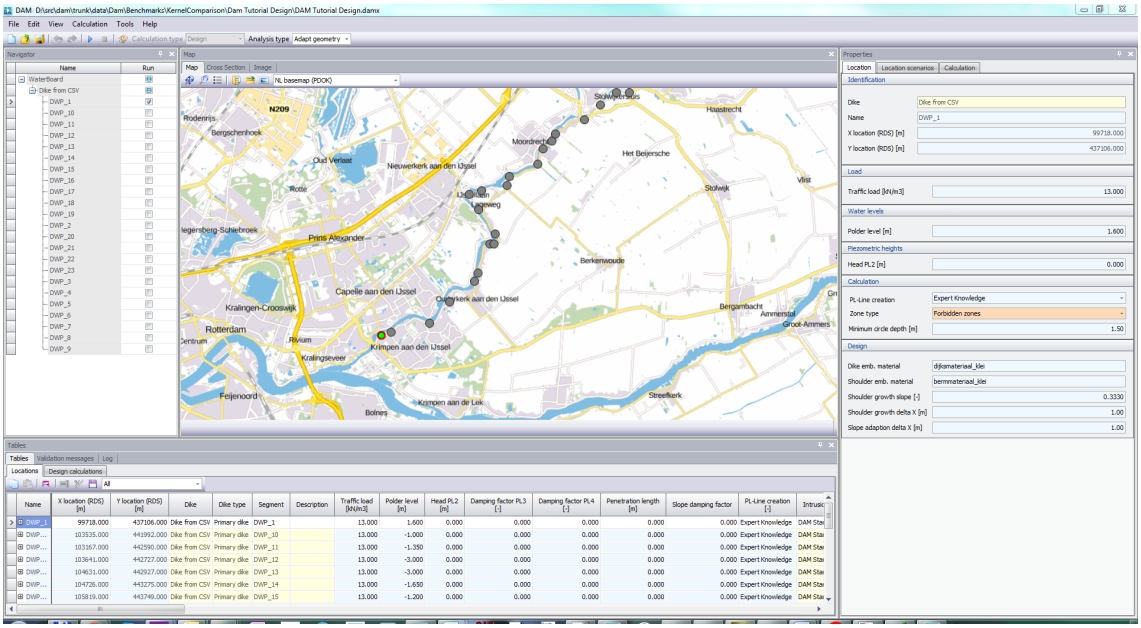

Figure 2 DAM User Interface.

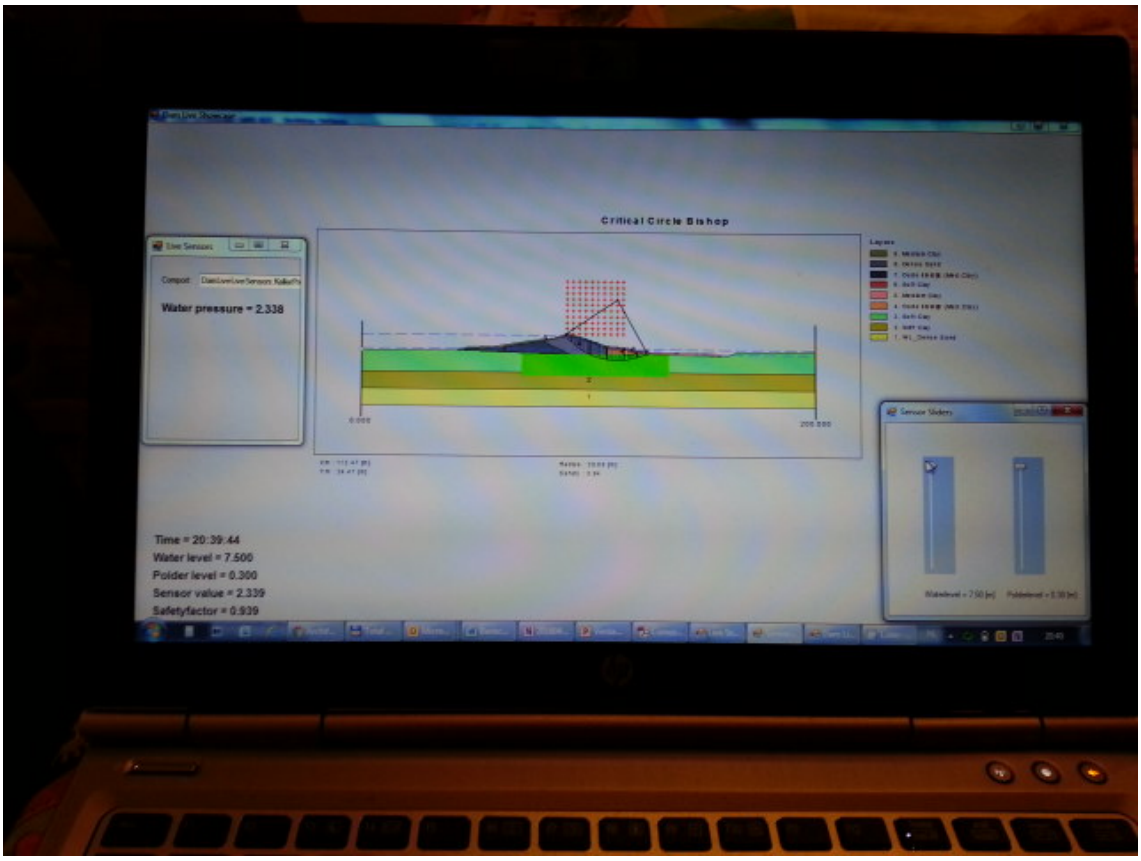

Figure 3 DAM Live Showcase User Interface.

Figure 4 DAM Live sensor.

These clients interact directly with the DAM Engine (see next section).

3.2 DAM Engine

The DAM Engine is the engine for the DAM computation. It contains several submodules, depending on which computation has to be made. As can be seen in Figure 1 the submodules are

- Dikes assessment
- Dikes assessment regional
- Dikes operational (for realtime calculations)
- Dikes design
- Dikes NWO (Niet Waterkerende Objecten)

Depending on the client, 1 or more of these submodules can be addressed by the client. The DAM Engine has no knowledge of the clients that use the Engine and there will be no code dependencies from the Engine to the clients.

3.3 Failure mechanisms

The DAM Engine uses external failure mechanisms. These are completely independent, and have no knowledge of the DAM Engine. So there will be no code dependencies from the failure mechanisms to the DAM Engine. Furthermore development, maintenance and support of these failure mechanisms are not in the scope of the DAM system.

4 Architectural Choices

4.1 Module dependencies

As can be seen in Figure 1 the arrows pointing between the main parts of the system are only 1 way. This means that e.g. the DAM clients may address the DAM Engine, but the DAM Engine may not address the DAM clients.

4.2 External libraries and components

DAM uses third-party libraries and components. Only open sources and free components, that are free to redistribute, are allowed to be used.

Furthermore DAM uses the Delta Shell Light (DSL) library, that is developed by Deltares.

Due to choices that have been made in the past, the UI modules of DSL use DevExpress, which is a commercial library. Free redistribution of the DevExpress is allowed by the development license that is used by Deltares.

In the future the dependency on DevExpress should be removed, e.g. by using other, open source, UI libraries.

In the next sections the libraries that are used by the components are summarized.

4.3 DAM UI (DSL)

This client uses the full Delta Shell Light (DSL) library
- DSL-Core: standard library with general common functionality
- DSL-Probabilistic: probabilistic functionality
- DSL-Geographic: GIS functionality
- DSL-Geo: geotechnics functionality
- DSL-GeoIO: geotechnics import and database functionality
- DSL-FormsStandard: standard UI functionality
- DSL-FormsMaps: extends FormsStandard with GIS functionality
- DSL-FormsGeo: extends FormsStandard with geotechnical functionality

Other libraries that are used are
- Dot Spatial: GIS library
- Commandline Parser: library for parsing commandline options
- Lumenworks: CSV import library
- SQLite: SQLite database access library
- Firebird: Firebird database access library

4.4 DAM Live

DAM Live only uses part (non-UI modules) of the DSL library
- DSL-Core
- DSL-Geographic

4.5 DAM Live Showcase

DAM Live only uses part (non-UI modules) of the DSL library
- DSL-Core

4.6 DAM Kernel Comparison Runner

DAM Kernel Comparison Runner only uses part (non-UI modules) of the DSL library
- DSL-Core

4.7 DAM Engine

The DAM Engine only uses part (non-UI modules) of the DSL library
- DSL-Core
- DSL-Probabilistic
- DSL-Geo

Other libraries that are used are
- Math.Net: mathematical library

4.8 Failure mechanisms

The failure mechanisms are completely independent, and will incorportate their own libraries. The only restriction that is imposed on the failure mechanisms is that, if they share the same libraries with DAM, the libraries should have the same version. The following libraries are know to be used by some failure mechanisms:
- DSL-Core
- DSL-Probabilistic
- DSL-Geographic
- DSL-Geo
- DSL-GeoIO
- Math.Net

5 Version control

As version control system Subversion with the Tortoise client will be used. The layout of the SVN repository will reflect the components of DAM as shown in Figure 1.

The failure mechanisms are not part of DAM itself and thus not of the DAM repository. Instead, the failure mechanisms will be stored in their own repositories. DAM will refer to the failure mechanisms as external libraries.

5.1 DAM repository main layout

In Figure 5 the main layout is shown.

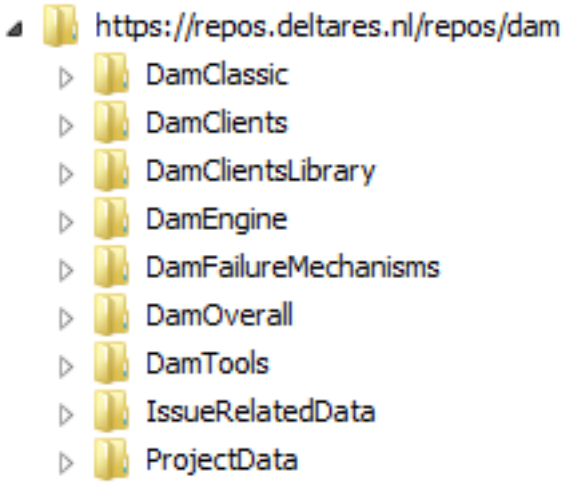

Figure 5 DAM SVN main layout.

The 5 main parts are
- DamClients - the client applications that use the DAM Engine.
- DamClientsLibrary - shared libraries by the DAM clients.
- DamFailureMechanisms - the failure mechanisms that were used in the orginalversion of DAM; they will also be used in the DAM Engine.
- DamEngine - the computational engine of DAM.
- DamOverall - general documentation for DAM (like this document).

The maps "'DamClientsLibrary"', "'DamEngine"' and "'DamOverall"' have their own trunk/branches/tags layout.

The remaining parts are
- DamClassic - the orginal (15.1) version of DAM (which has UI and calculation engine in one solution); it has its own trunk/branches/tags layout.
- DamTools - independent tools that are not part of DAM itself, but support the work processes of DAM (e.g. Dam Edit Design).
- IssueRelatedData - data which belong to Jira issues; DAM uses Jira as issuetracking system, which supports attachments, but DAM attachments can be

very large, so it is better to save the attachments in this location; the attachments are stored in a map with the name of the issue number (e.g. MWDAM-982)
- ProjectData - used for archiving DAM project data.

5.2 DAM repository clients layout

Each DAM client has its own entry in the clients map. In Figure 6 the layout of the DAM clients map is shown.

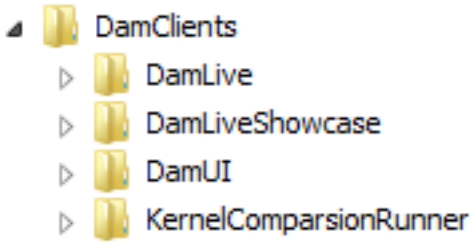

Figure 6 DAM SVN clients layout.

Each of the "'dam clients"' has its own trunk/branches/tags layout.

The currently known applications are
- Dam UI - the current DAM desktop application.
- DamLive - the current runner for the Fews operational system.
- DamLiveShowcase - a demo application to show DamLive with live sensors.
- KernelComparisonRunner - a commandline utility for comparing the results of different macrostability kernels.

5.3 DAM repository clients library layout

In Figure 7 the layout of the DAMUI clients library map is shown.

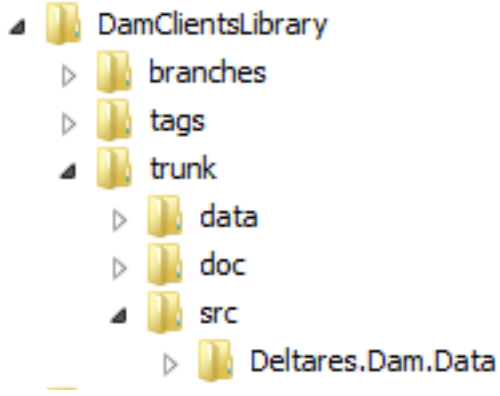

Figure 7 DAM SVN clients library layout.

At this moment only one project is foreseen to be put in the "'dam clients library"' src map is:
- Deltares.Dam.Data: this map contains the business layer.

5.4 DAM repository DAM Engine

In Figure 8 the layout of the DAM Engine map is shown.

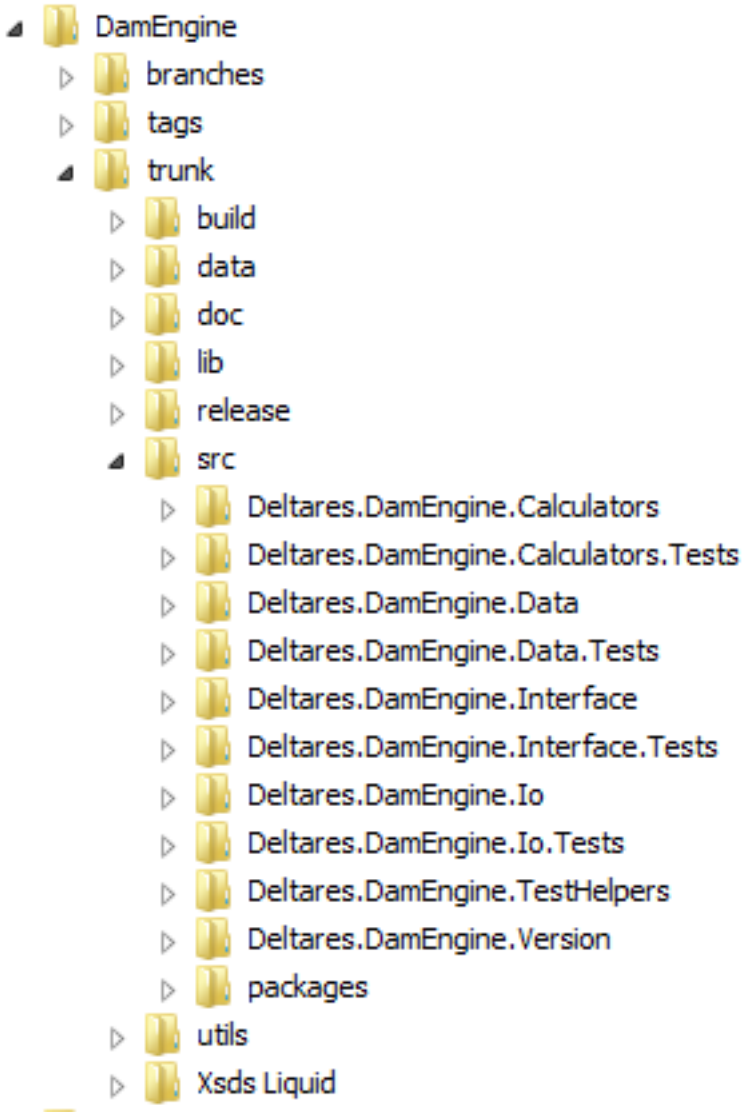

Figure 8 DAM Engine layout.

At this moment the projects foreseen to be put in the "'dam engine"' src map are:

- Deltares.DamEngine.Calculators: this map contains classes to perform the actual calculations for the DAM Engine.
- Deltares.DamEngine.Data: this map contains pure data classes.
- Deltares.DamEngine.Interface: this map contains classes to interface to the engine.
- Deltares.DamEngine.Io: this map contains classes performing Input/Output to and from the engine.
- Deltares.DamEngine.Version: this map contains a class to configure the version number for all classes.

Each project will have a corresponding test
0y99o9ogh==igiii99i9ki9999999999og9iioiiioiiii=ctionDAM repository Failure

Mechanisms

The original DAM implementation had its own failure mechanisms implemented. In the current architecture, the failure mechanisms are not part of the DAM engine itself. They will be implemented as independent kernels. In Figure 8 the layout of the DAM Failure Mechanisms map is shown.

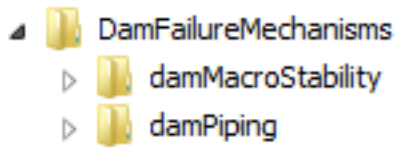

Figure 9 DAM Failure Mechanisms layout.

As will be the case with other, third party kernels, a wrapper will be written around these kernels, to use them in the DAM engine. These wrappers are a part of the DAM engine.

At this moment the following failure mechanisms will be put here:

- Macrostability: this is an implementation of the Delphi D-Geo Stability kernel.
- Piping: there are 3 different piping methods that are implemented here:
  - Bligh
  - Sellmeijer 4 forces
  - Sellmeijer VNK (neural network)

5.5 DAM repository full layout

The full layout of the DAM repository will be as shown in Figure 10.

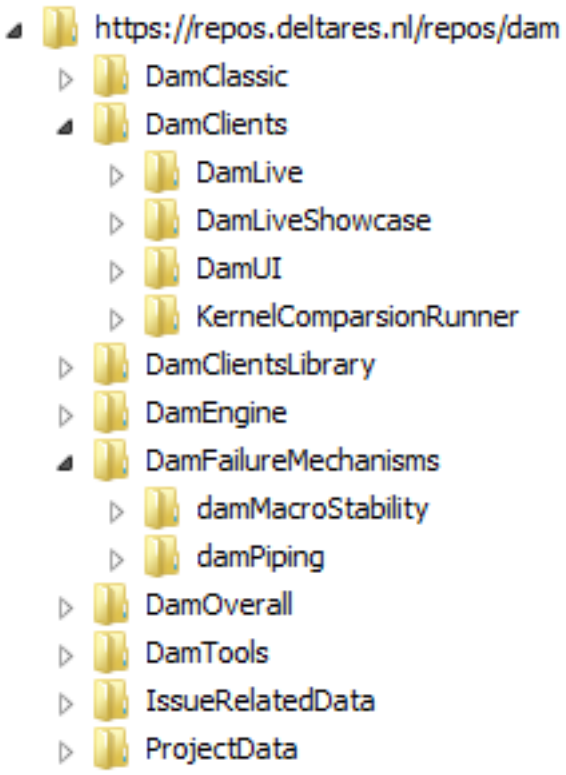

Figure 10 DAM SVN full layout.

6 Releasing a product

This is a description of the steps needed to create a DAM release. An external release is normally a release of a DAM client (DAM UI or DAM Live). The order of releasing the several modules of the DAM system is important because of the dependencies between the several modules. So keep the order of releasing as is specified in this chapter.
It is not always necessary to release all modules, but a module needs to be released, when a dependent module has been released.

6.1 Dam Failure Mechanisms

This chapter describes the failure mechanisms which are specifically developed for Dam.

6.1.1 DamMacroStability

This failure mechanism is a wrapper around the D-Geo Stability executable (part of the D-Serie). The following steps are needed to release this failure mechanism.

- Update to the correct version of DGeoStability.exe (to be found in lib\Deltares\DGeoStability)
- Update version number in .\src\Deltares.DamMacroStability.Version\GlobalAssembly.cs.svn
- Release binary artifacts (from Teamcity) to the directory .\release

6.1.2 DamPiping

The following steps are needed to release this failure mechanism.

- Update version number in .\src\Deltares.DamPiping.Version\GlobalAssembly.cs.svn
- Release binary artifacts (from Teamcity) to the directory .\release

6.2 DGSMStabDam

This module is responsible for creating the D-Geo Stability project files. It is used by the DAM Engine. The module is written in Delphi and the sources can be found in another repository:
https://repos.deltares.nl/repos/pvcsprojects/ApplicationSupport/DAM/DGSMStabDam

The following steps are needed to release this module.

- Update version number in .\src\DGSMStabDAM\ResourceVersionInfo.rc.svn
- Release binary artifacts (from Teamcity) to the directory .\release

6.3 Dam Engine

This module has the following dependencies:

- DAM Failure Mechanisms
- DGSMStabDam

The following steps are needed to release this module.

- Make sure the following external points to the correct tag/branch: lib\DGSMStabDam
- Make sure the following external points to the correct tag/branch: lib\FailureMechanisms\DamMacroStability
- Make sure the following external points to the correct tag/branch: lib\FailureMechanisms\DamPiping
- Make sure the following failure mechanism contains the correct version: lib\FailureMechanisms\WtiPipingMerged
- Update version number in .\src\Deltares.DamEngine.Version\GlobalAssembly.cs.svn
- When the (normal) build is finished on team city, start in Team City the Dam Release, Dam Engine project by hand to trigger the signing project on Team City.
- Download the binary artifacts (from the signing project from Teamcity) and add them to to the directory .\release
- Create a tag for the version.
- Reset the third version number in .\src\Deltares.DamEngine.Version\GlobalAssembly.cs.svn to 0 for the Engine project in the branch.

6.4 Dam UI

This application is dependent on the DAM Engine.
The following steps are needed to release this application.

- Make sure the following folder contains the correct version of the licensing system: lib\Authorization
- Make sure the following folder contains the correct version of DevExpress: lib\DevExpress
- Make sure the following externals points to the correct tag/branch: lib\DamEngine
- Make sure the following externals points to the correct tag/branch: lib\DSL-Core
- Make sure the following externals points to the correct tag/branch: lib\DSL-FormsGeo
- Make sure the following externals points to the correct tag/branch: lib\DSL-FormsMap
- Make sure the following externals points to the correct tag/branch: lib\DSL-FormsStandard
- Make sure the following externals points to the correct tag/branch: lib\DSL-Geo
- Make sure the following externals points to the correct tag/branch: lib\DSL-Geographic
- Make sure the following externals points to the correct tag/branch: lib\DSL-GeoIO
- Make sure the following externals points to the correct tag/branch: lib\DSL-Probabilistic
- Update version number in .\src\Deltares.Dam.Version\GlobalAssembly.cs.svn
- Update version number in .\src\DamClientsLibrary\DamClientsLibrary.Version\GlobalAssembly.cs.svn
- When the (normal) build is finished on team city, start in Team City the Dam Release, Dam UI project by hand to trigger the signing project on Team City.
- Download the binary artifacts (from the signing project from Teamcity) and add

them to to the directory .\release\InstallVersion.
- Update the version number in Product.wxs in the Setup project to the correct version.
- When the (normal) build of Dam UI Setup is finished on team city, start in Team City the Dam Release, Dam UI Setup project by hand to trigger the signing project on Team City.
- Download the binary artifacts (from the Signing_DamUISetup project from Teamcity) and add them to to the directory .\release.
- Create a tag for the version.
- Reset the third version number to 0 in .\src\Deltares.Dam.Version\GlobalAssembly.cs.svn
- Reset the third version number to 0 in .\src\DamClientsLibrary\DamClientsLibrary.Version\GlobalAssembly.cs.svn
- Reset the third version number to 0 in Product.wxs in the Setup project.

6.5 DamLive

This application is dependent on the DAM Engine.
The following steps are needed to release this application.

- Make sure the following folder contains the correct version of the licensing system: lib\Authorization
- Make sure the following externals points to the correct tag/branch: lib\DamEngine
- Make sure the following externals points to the correct tag/branch: lib\DSL-Core
- Make sure the following externals points to the correct tag/branch: lib\DSL-Geo
- Make sure the following externals points to the correct tag/branch: lib\DSL-Geographic
- Make sure the following externals points to the correct tag/branch: lib\DSL-GeoIO
- Make sure the following externals points to the correct tag/branch: lib\DSL-Probabilistic
- Update version number in .\src\Deltares.DamLive.Version\GlobalAssembly.cs.svn
- Release binary artifacts (from Teamcity) to the directory .\release

7 Literature