



Automating GeMS Submissions from an Enterprise Geodatabase



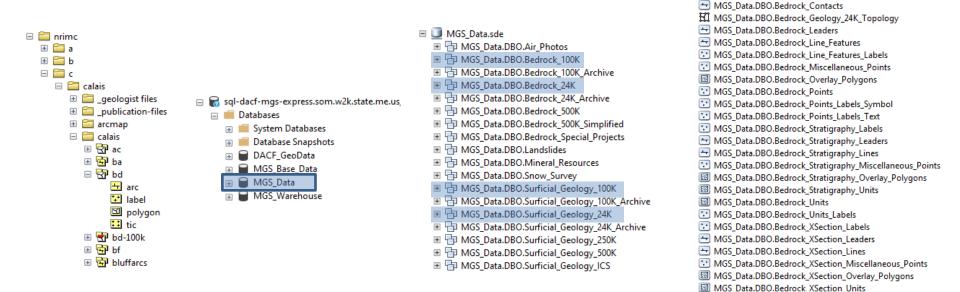
Christian Halsted
Director, Earth Resources Information
Maine Geological Survey
Christian.H.Halsted@Maine.gov



☐ ☐ MGS Data.DBO.Bedrock 24K

How Did We Get Here?

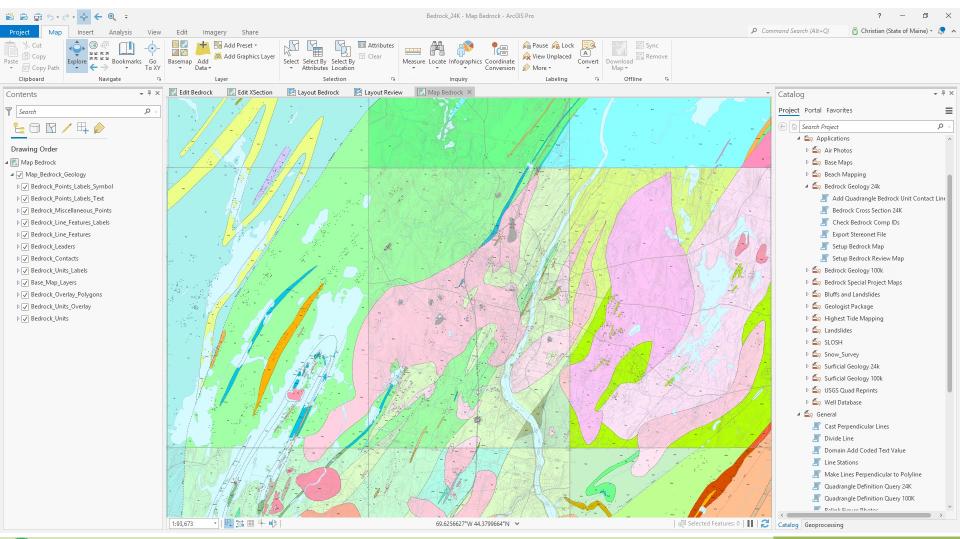
- 1996: Single quadrangle-based mapping with ArcInfo Coverages
- 2014: SQL Server Express Enterprise Geodatabase for multi-quadrangle, multiscale, multi-map type
- 2017: NCGMP09
- 2018: MGS GeMS gap analysis
- 2020: StateMap Supplemental project GeMS workflow development





Working with an Enterprise Geodatabase in GIS

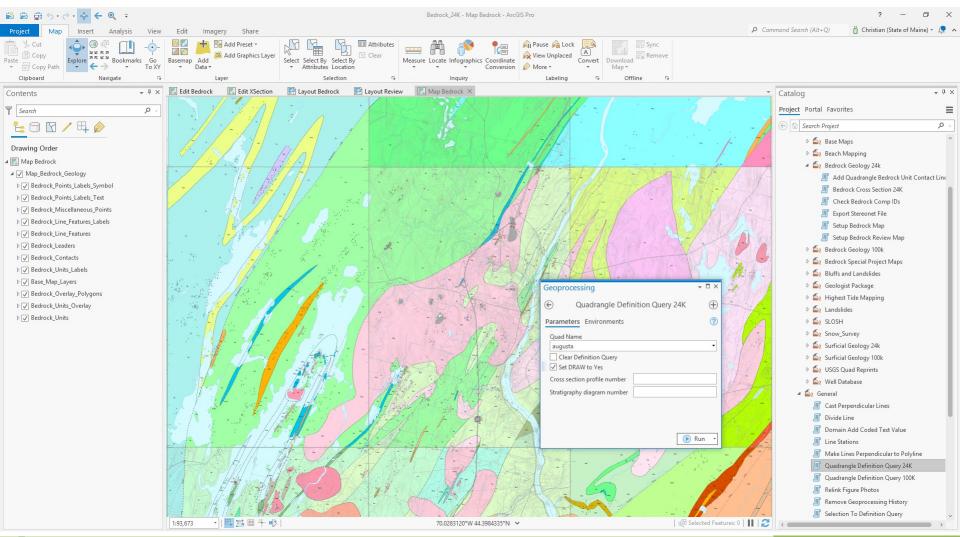
Features for all quads by map type and scale are stored in each feature class





Working with an Enterprise Geodatabase in GIS

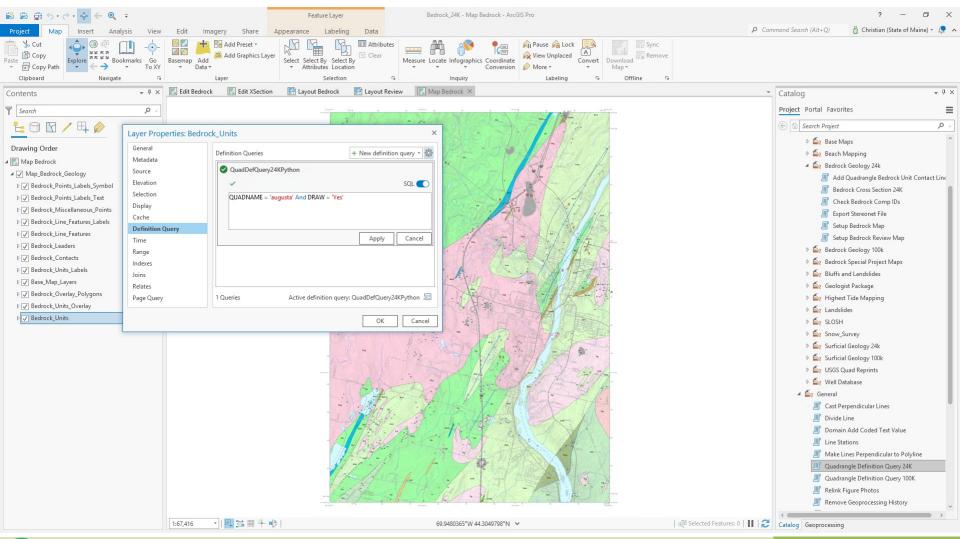
Custom Python tool applies definition query to all layers in the map





Working with an Enterprise Geodatabase in GIS

Only features in a single quad are now displayed for map production





Guiding Principles for Complying with GeMS

- Minimal disruption to current map production
- Minimal level of additional work by the mapping geologists and information management staff
- Treat GeMS as a data standard, not map standard
- Only GeMS Level 3 submissions
- Create a defined, scripted, documented workflow for GeMS conversion of bedrock and surficial maps at 24K and 100K scales
- GeMS is opportunity to improve MGS workflows around GeoLex, feature-level metadata, map text



What Was Missing?

Inventoried MGS feature classes to discover additional required GeMS attributes that needed to be captured.

GeMS Field	MGS Feature Class
IDENTITY_CONFIDENCE	 Bedrock_Contacts Bedrock_Line_Features Bedrock_Overlay_Polygons Bedrock_Units Bedrock_XSection_Lines Bedrock_XSection_Overlay_Polygons Bedrock_XSection_Units tbl_Bedrock_Feature
EXISTENCE_CONFIDENCE	Bedrock_ContactsBedrock_Line_FeaturesBedrock_XSection_Lines
LOCATION_CONFIDENCE	 Bedrock_Contacts Bedrock_Line_Features Bedrock_XSection_Lines tbl_Bedrock_Points
ORIENTATION_CONFIDENCE	• tbl_Bedrock_Feature
LOCATION_METHOD	• tbl_Bedrock_Points

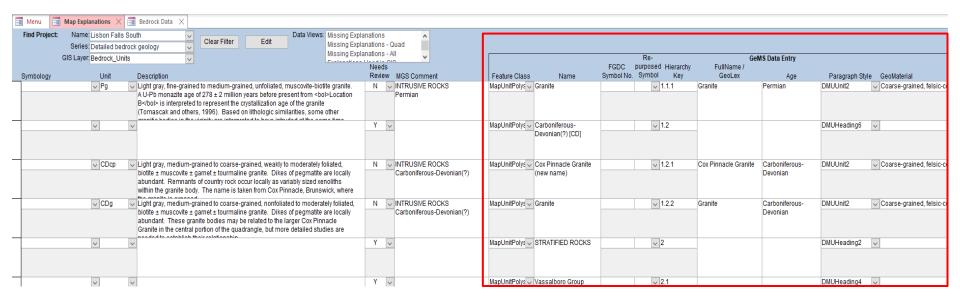
List of additional fields needed for GeMS compliance in Bedrock 24K mapping feature classes and tables.



What Was Missing?

MGS map explanation table needed a number of new fields to control GeMS destination feature classes and description of map units.

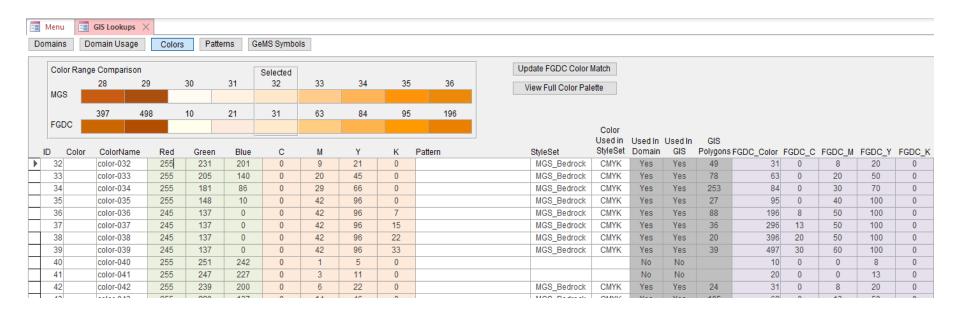
Example of GeMS fields added to the MGS Explanation table (red box).





Reference Tables

Reference tables for geomaterials, paragraph styles, and FGDC colors, symbols, and patterns were created in the MGS database to control data entry. In the case of colors and patterns, these tables also provide a crosswalk between the values traditionally used by MGS and the FGDC values.

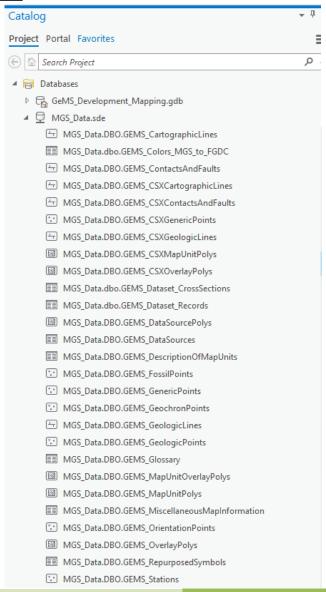




GeMS Staging Tables

The GeMS Create New Database tool was used to create a file geodatabase with all the default tables and feature classes. These objects were imported into the MGS enterprise geodatabase to serve as staging tables/feature classes for GeMS data.

Field data types weren't always correctly transformed between file and enterprise geodatabase.





Data Mapping

The existing MGS feature classes and tables, or subsets of features, were mapped to the required GeMS feature classes and tables.

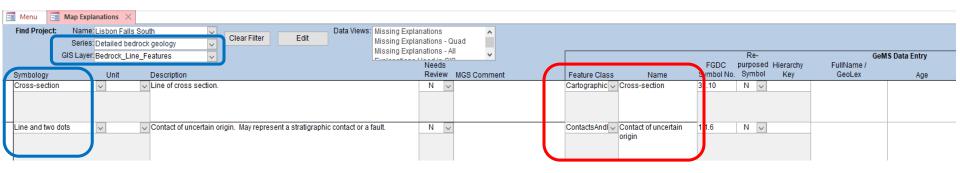
MGS Feature Class (criteria)	GeMS Feature Class	
Units	MapUnitPolys	
Contacts (not overlapped by Bedrock_Lines)	ContactsAndFaults	
Lines (with GEMS_FeatureClass = 'ContactsAndFaults')		
Units (dissolved units to make map boundary)		
Points	Stations	
Points_Labels_Symbol (with GEMS_FeatureClass = 'OrientationPoints')	OrientationPoints	
Points_Labels_Symbol (with GEMS_FeatureClass = 'GeologicPoints')	GeologicPoints	
Units_Labels	GenericPoints	
Line_Features_Labels		
Miscellaneous_Points (SYMBOLOGY <> 'Geochronology point')		
Line_Features (GEMS_FeatureClass = 'GeologicLines')	GeologicLines	
Line_Features (GEMS_FeatureClass = 'CartographicLines')	CartographicLines	
Leaders		
Units (SYMBOLOGY_OVERLAY IS NOT NULL)	MapUnitOverlayPolys	
Overlay_Polygons	OverlayPolys	
Quadrangle_Split_Authors	DataSourcePolys	
tbl_Publications	Miscellaneous Map Information	
XSection_Units	CSXMapUnitPolys	
XSection_Lines (with GEMS_FeatureClass = 'CSXContactsAndFaults')	CSXContactsAndFaults	
XSection_Overlay_Polygons	CSXOverlayPolys	
XSection_Lines (with GEMS_FeatureClass = 'CSXCartographicLines')	CSXCartographicLines	
XSection_Leaders		
XSection_Lines (with GEMS_FeatureClass = 'CSXGeologicLines')	CSXGeologicLines	
XSection_Labels	CSXGenericPoints	
XSection_Miscellaneous_Points		



Data Mapping

Individual feature types in existing MGS feature classes can be mapped to their GeMS destination feature class using the GeMS:Feature Class field in the MGS Map Explanation table.

Example of different feature types in a single MGS feature class being mapped to two different GeMS feature classes.



MGS source feature class and feature types

GeMS destination feature class and feature types



GeMS Build Process Steps

Process Step	Manual	MGS Tool	GeMS Tool
Check topology on MGS Data	fixes	Χ	
Copy data from MGS to GeMS tables		Х	
Create GeMS folder structure		X	
Export GeMS tables to a file gdb		X	
Create metadata XML file		X	
Validate metadata with USGS mp (optional)	X	Planned	
Load metadata to fgdb		X	
Copy GeMS Checklist PDF file to quad folder		X	
Run Validate Database			X
Delete Validation fgdb and unneeded validation files		In progress	
Run Translate to Shapefiles		Planned	X
Run Geologic Names Check			X
Fill out Geologic Names file	X		
Copy map PDF into quad folder		X	
Copy map mxd into /publication folder	X	Planned	
Repoint disconnected layers in map mxd	X	Planned	
Copy map text, photos, figures, captions, references into /resources folder	X	In progress	
Copy fonts and style files into /resources folder		X	
Create transmittal letter PDF		Х	
Fill out GeMS Checklist PDF	Х		
Create GeMS quad zip file	Х	Planned	



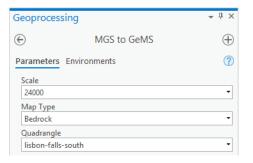
GeMS Build Process Steps

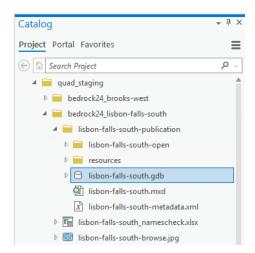
Process Step	Manual	MGS Tool	GeMS Tool
Check topology on MGS Data	fixes	Х	
Copy data from MGS to GeMS tables		Х	
Create GeMS folder structure		X	
Export GeMS tables to a file gdb		X	
Create metadata XML file		X	
Validate metadata with USGS mp (optional)		X	
Load metadata to fgdb		X	
Copy GeMS Checklist PDF file to quad folder		X	
Run Validate Database			X
Delete Validation fgdb and unneeded validation files		X	
Run Translate to Shapefiles		X	
Run Geologic Names Check			X
Fill out Geologic Names file	X		
Copy map PDF into quad folder		X	
Copy map mxd into /publication folder		X	
Repoint disconnected layers in map mxd		X	
Copy map text, photos, figures, captions, references into /resources folder		X	
Copy fonts and style files into /resources folder		X	
Create transmittal letter PDF		X	
Fill out GeMS Checklist PDF	X		
Create GeMS quad zip file		X	



MGS GeMS Automation Tools

- MGS tools are a combination of:
 - Database stored procedures that extract data from MGS database into staging tables and perform some validation
 - Python scripts that
 - Call stored procedures
 - Create folders and copy files with native Python
 - MS Access VBA that
 - Call stored procedures for data review
 - Runs reports
- All tools execute for the selected quadrangle, scale (24K or 100K), and map type (bedrock or surficial)
- This provides the flexibility to create GeMS submissions for any map published by MGS.







MGS GeMS Automation Costs & Benefits

Costs

- Code maintenance
 - SQL Server (Example: stored procedure for staging data is 355 SELECT statements ~3200 lines of code)
 - New MGS Python toolbox
 - New or revised MS Access forms and reports
- Changes to the GeMS standard will create work for MGS to revise tools
- Dependency on broad IT platform and IT staff

Benefits

- GeMS file preparation by mappers, not GIS/IT staff
- Consistency in submission
- Time savings
- Recover costs of data entry
- Data driven smaller risk from vendor software change

```
MGS to GeMS.tbx
                                                                             MGS to GeMS
TRUNCATE TABLE GEMS_MapUnitPolys;
INSERT INTO GEMS_MapUnitPolys
                                                                              MGS Topology Check
     ,[MapUnit]
     ,[IdentityConfidence]
     [Label]
     [DataSourceID]
     [Notes]
     ,[Shape]
,[Symbol_MGS]
            --'C#'=cambrian, 'T#'=triassic, 'M#'=mesozoic, 'P#'=pennsylvanian, 'Cz#'=cenozoic 'Pz#'=paleozoic
            AS MapUnit
         ,IDENTITY_CONFIDENCE AS IdentityConfidence
         ,CAST(tbl GEMS Colors MGS to FGDC.FGDC Color AS nvarchar(50)) AS Symbol
         ,SYMBOLOGY AS Symbol_MGS
         (SELECT DataSources ID FROM GEMS DataSources WHERE SOURCE = 'this report') AS DataSourceID
         ,GEMS_NOTES AS Notes
          'MUP' + CAST(Bedrock Units evw.OBJECTID AS nvarchar(10)) AS MapUnitPolys ID --COMPID?
         ,Bedrock_Units_evw.QUADNAME
         , 'Bedrock' AS Map
    FROM Bedrock_Units_evw
        JOIN tbl GEMS Colors MGS to FGDC ON Bedrock_Units_evw.SYMBOLOGY = tbl_GEMS_Colors_MGS_to_FGDC.ColorName
     WHERE Bedrock_Units_evw.QUADNAME = @Quad
    UNION ALL
SELECT
        ,IDENTITY_CONFIDENCE AS IdentityConfidence
         ,CAST(tbl_GEMS_Colors_MGS_to_FGDC.FGDC_Color AS nvarchar(50)) AS Symbol
         (SELECT DataSources_ID FROM GEMS_DataSources WHERE SOURCE = 'this report') AS DataSourceID
               + CAST(Surficial Geology Units evw.OBJECTID AS nvarchar(10)) AS MapUnitPolys ID --COMPID??
         , Surficial_Geology_Units_evw.QUADNAME
         .'Surficial' AS Map
    FROM Surficial_Geology_Units_evw
         JOIN tbl_GEMS_Colors_MGS_to_FGDC_ON_Surficial_Geology_Units_evw.UNIT = tbl_GEMS_Colors_MGS_to_FGDC.ColorName
     WHERE Surficial_Geology_Units_evw.QUADNAME = @Quad
        REPLACE(REPLACE(REPLACE(REPLACE(REPLACE(WNIT,'C#','C'),'T#','T'),'M#','M'),'P#','P'),'Cz#','Cz'),'Pz#','Pz'
             --'C#'=cambrian, 'T#'=triassic, 'M#'=mesozoic, 'P#'=pennsylvanian, 'Cz#'=cenozoic 'Pz#'=paleozoic
         ,IDENTITY CONFIDENCE AS IdentityConfidence
          CAST(tbl_GEMS_Colors_MGS_to_FGDC.FGDC_Color AS nvarchar(50)) AS Symbol
        ,SYMBOLOGY AS Symbol_MGS
,(SELECT DataSources ID FROM GEMS_DataSources WHERE SOURCE = 'this report') AS DataSourceID
          . MUP + CAST(Bedrock_100K_Units_evw.OBJECTID AS nvarchar(10)) AS MapUnitPolys_ID --COMPID?
         ,Bedrock_100K_Units_evw.QUADNAME
         , 'Bedrock' AS Map
.'100000' AS Scale
     FROM Bedrock_100K_Units_evw
         JOIN tbl_GEMS_Colors_MGS_to_FGDC_ON_Bedrock_100K_Units_evw.SYMBOLOGY = tbl_GEMS_Colors_MGS_to_FGDC.ColorNa
     WHERE Bedrock 100K Units evw.QUADNAME = @Quad
```



Why Didn't We Just Transition to "regular" GeMS?

Because...

- Existing deep integration in a single system used by all staff
- File geodatabases would have been a major step backward

Examples of existing, pre-GeMS integration:

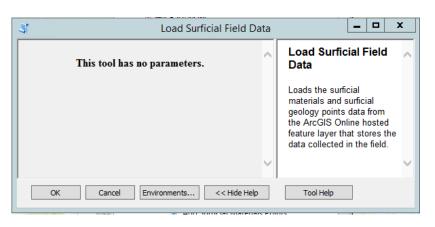
- Field data collection automation with MGS EGDB
- Bedrock Excel datasheet automation with MGS EGDB
- Map symbology functionality
- Map explanations (DMU) stored in the MGS EGDB
- MGS EGDB provides extent lat/long for Publications module
- Project Status module
- Citations module
- Grants module
- Photos module
- Web pages and web applications using services published from EGDB
- StateMap and Mapping Priorities dashboards linked to EGDB





Example: Field data collection automation with MGS EGDB

- iPad with Field Maps, Survey123, Photo app disconnected editing and basic navigation
- With cell signal, data continuously uploaded to ESRI cloud.
- Without cell signal, data stored on iPad and uploaded to ESRI cloud ASAP.
- Surficial Mapping:
 - Custom script brings data collected with Survey123 into MGS database.
 - Ready for map production.

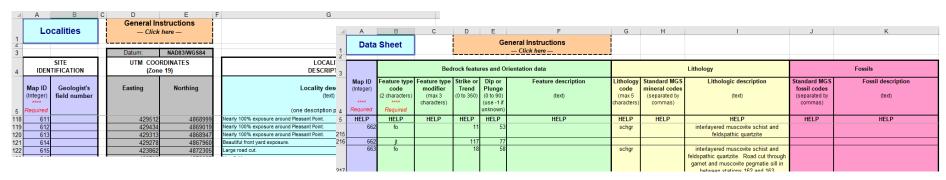


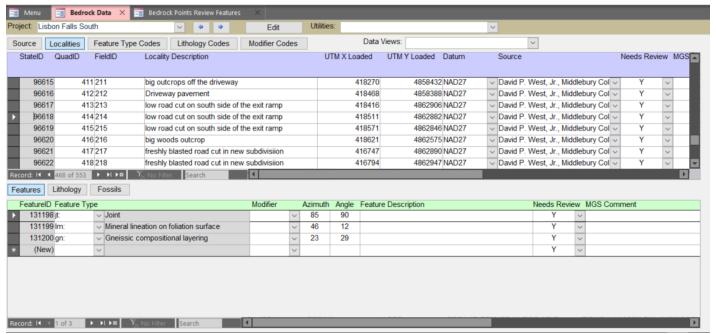




Example: Bedrock Excel datasheet automation with MGS EGDB

Tools and database triggers to load Excel datasheet and creating points in feature classes from non-spatial database tables

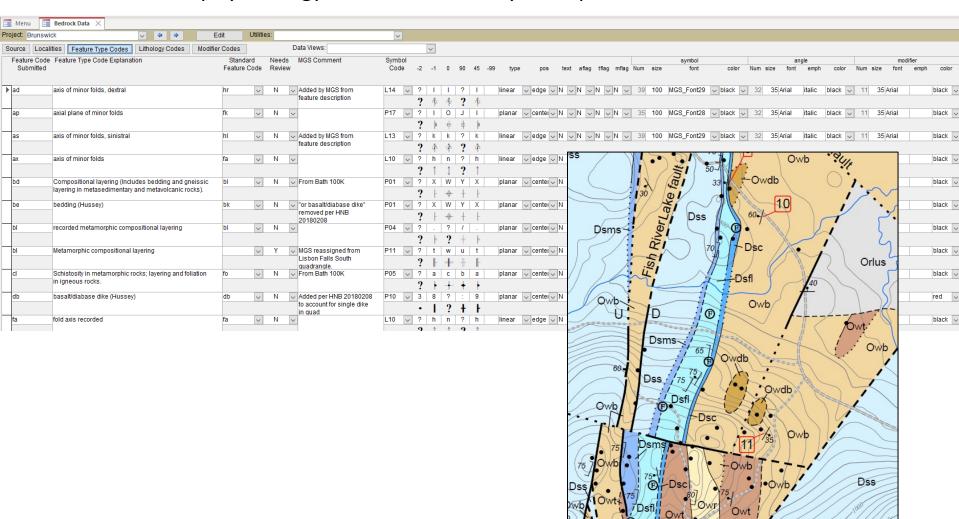






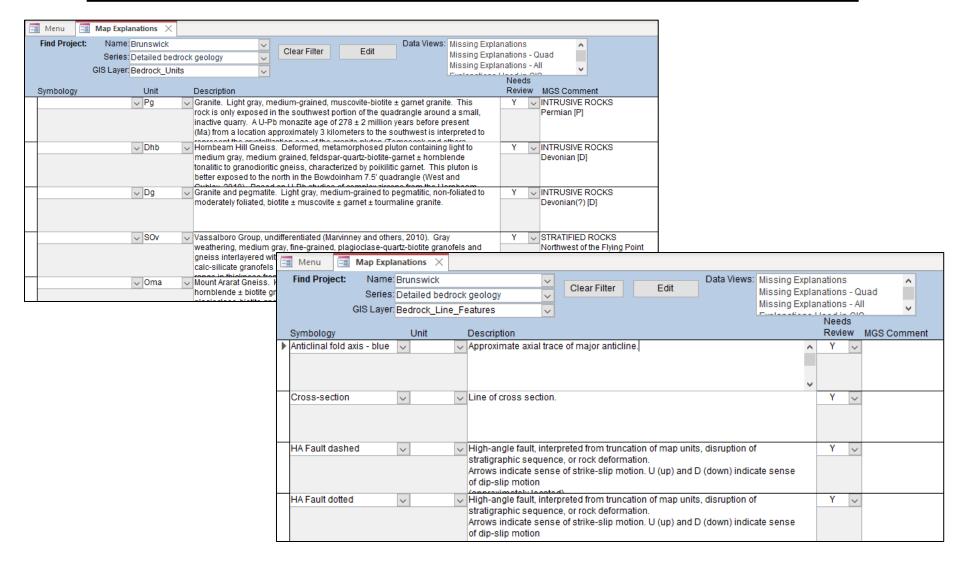
Example: Map symbology functionality

Map symbology controlled in GIS by non-spatial database tables





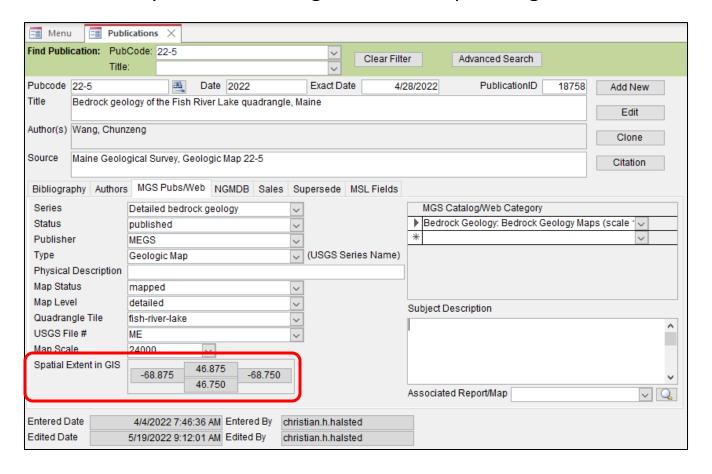
Example: Map explanations (DMU) stored in the MGS EGDB





Example: MGS EGDB provides extent Publications module

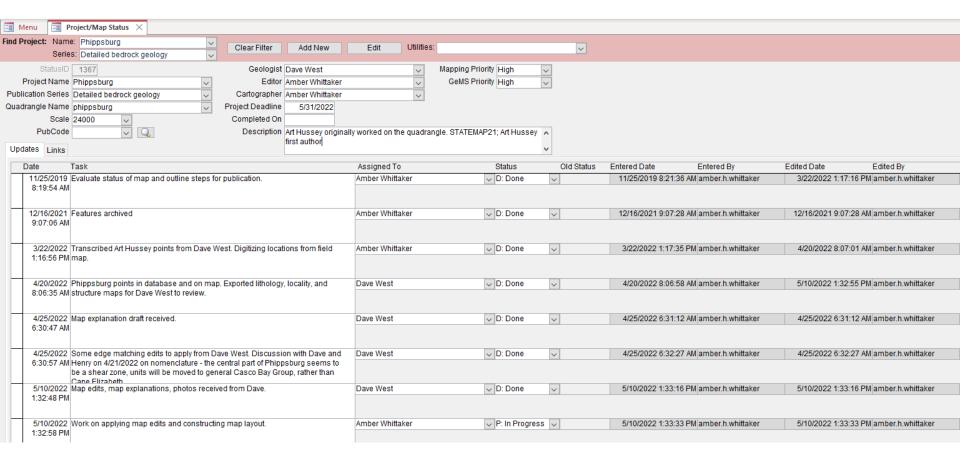
Automatically derived lat/long values from quadrangle feature class





Example: Project Status module

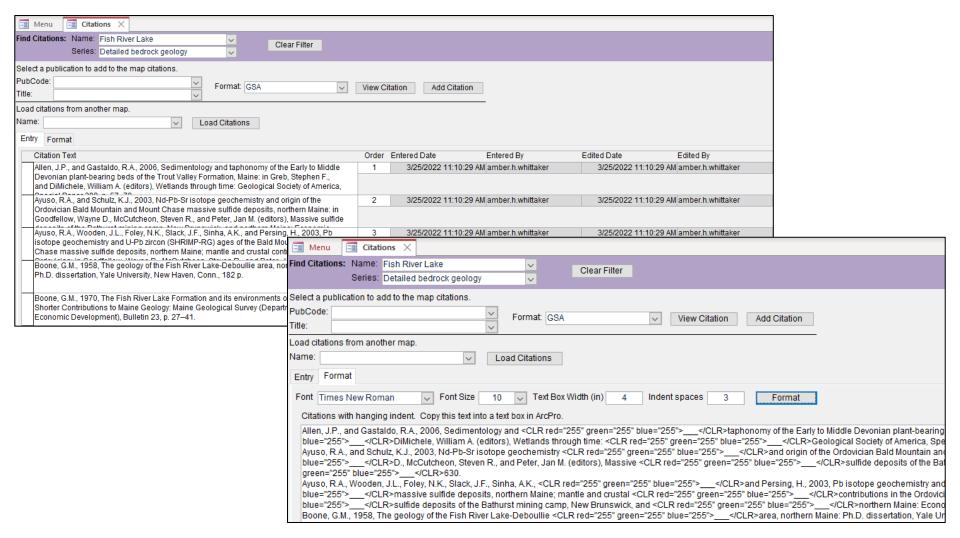
Tracks assignments, open issues, and completed tasks over the life of mapping project





Example: Citations module

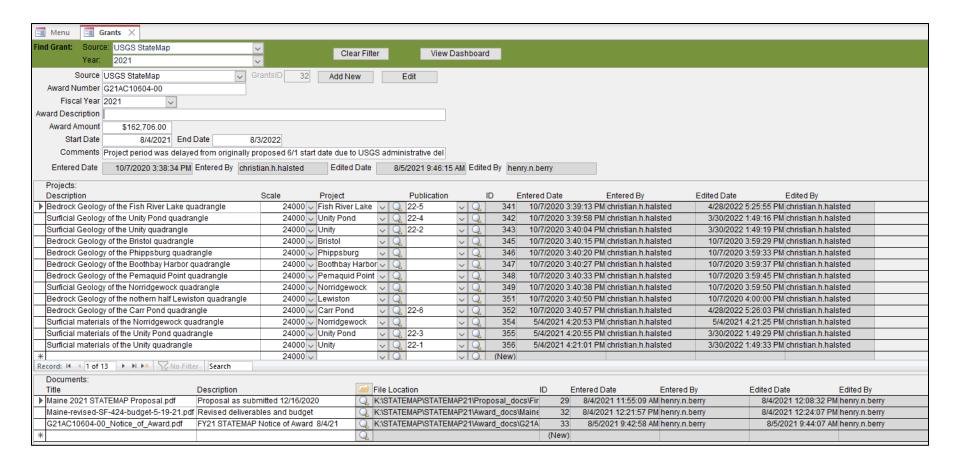
Map references entered, ordered, and formatted for map layout





Example: Grants module

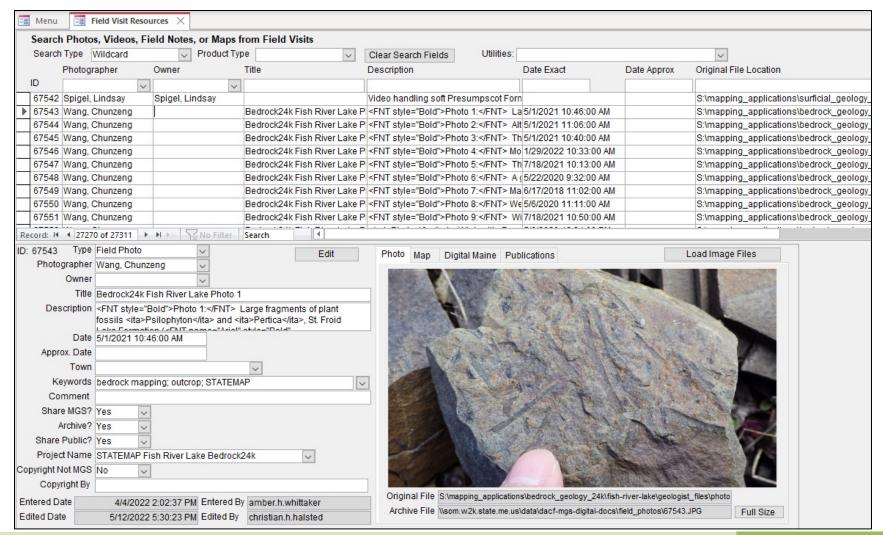
Tracks grant dates, grant documents, and maps due. Includes a link to the map in the Publications module when completed.





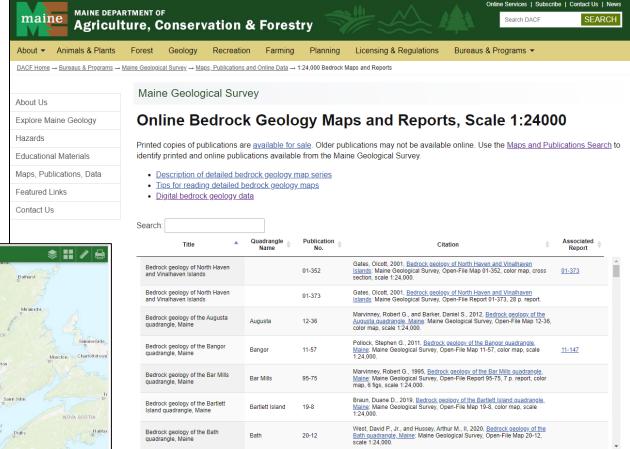
Example: Photos module

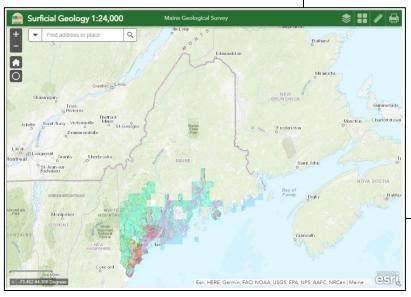
Tracks photos and their location that are used on the maps.



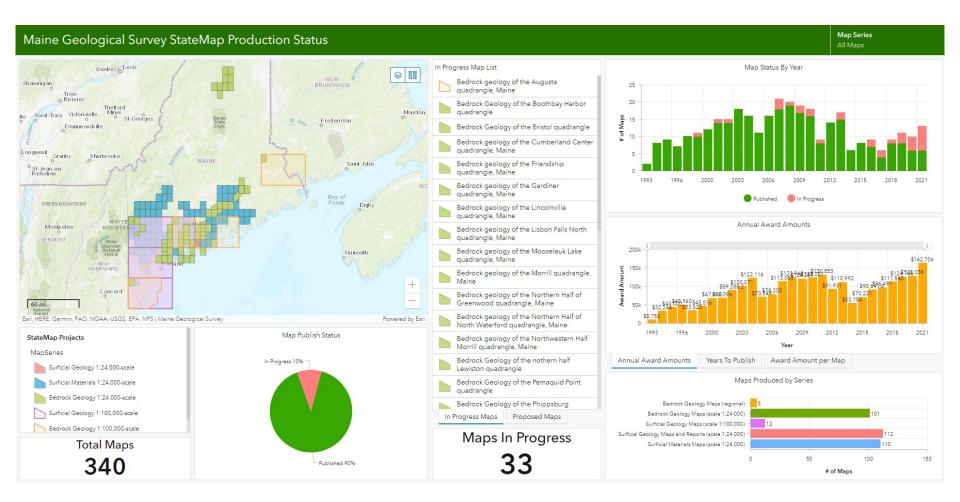


Example: Web pages and web apps using services from EGDB



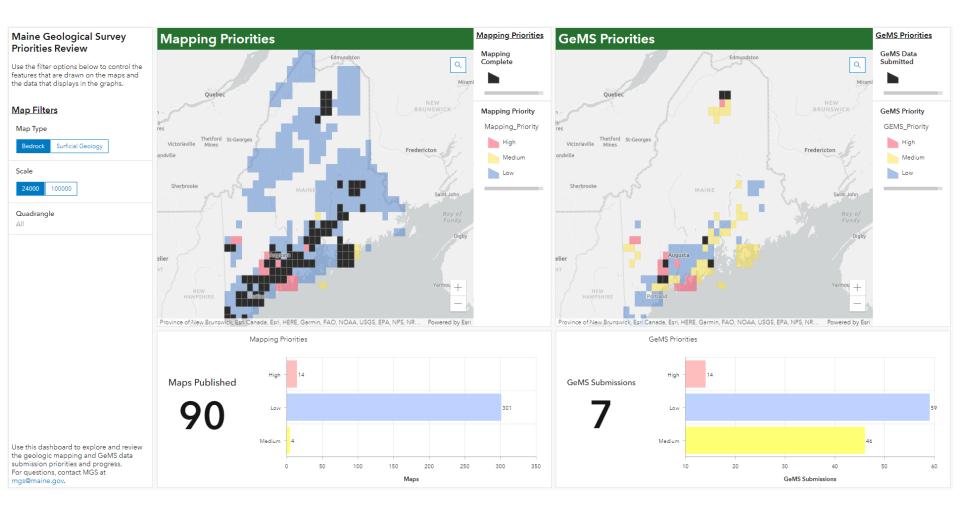


Example: Dashboards linked to EGDB





Example: Dashboards linked to EGDB





Possible Next Steps and Improvements

- Link database components to map layout photos, citations, map text, and references to automate more of map production
- Build contractor field data web mapping and data submission interface

Anyone interested in collaborating on:

- Incorporating some or all of the Validate Database tool into EGDB or updating tool to run against an EGDB
- Building GeMS web map service



Accomplishments and Conclusions

- Integration = consistency, repeatability, efficiency
- MGS to GeMS translation logic centralized in MGS database
- Geologic mapping staff trained in GeMS
- GeMS submission process fully documented
 - MGS Help document (19 pages)
 - GeMS Conversion Process Workflow (STATEMAP FTR)
- Eleven Level 3 and one Level 1 GeMS packages successfully submitted
- Costs quantified so improvements can be measured
 - 8-12 hours for author geologist to review and complete GeMS data entry on a published quadrangle (longer if not the author)
 - 6-10 hours for data manager to compile and validate GeMS submission package

Questions?

THANKS!

