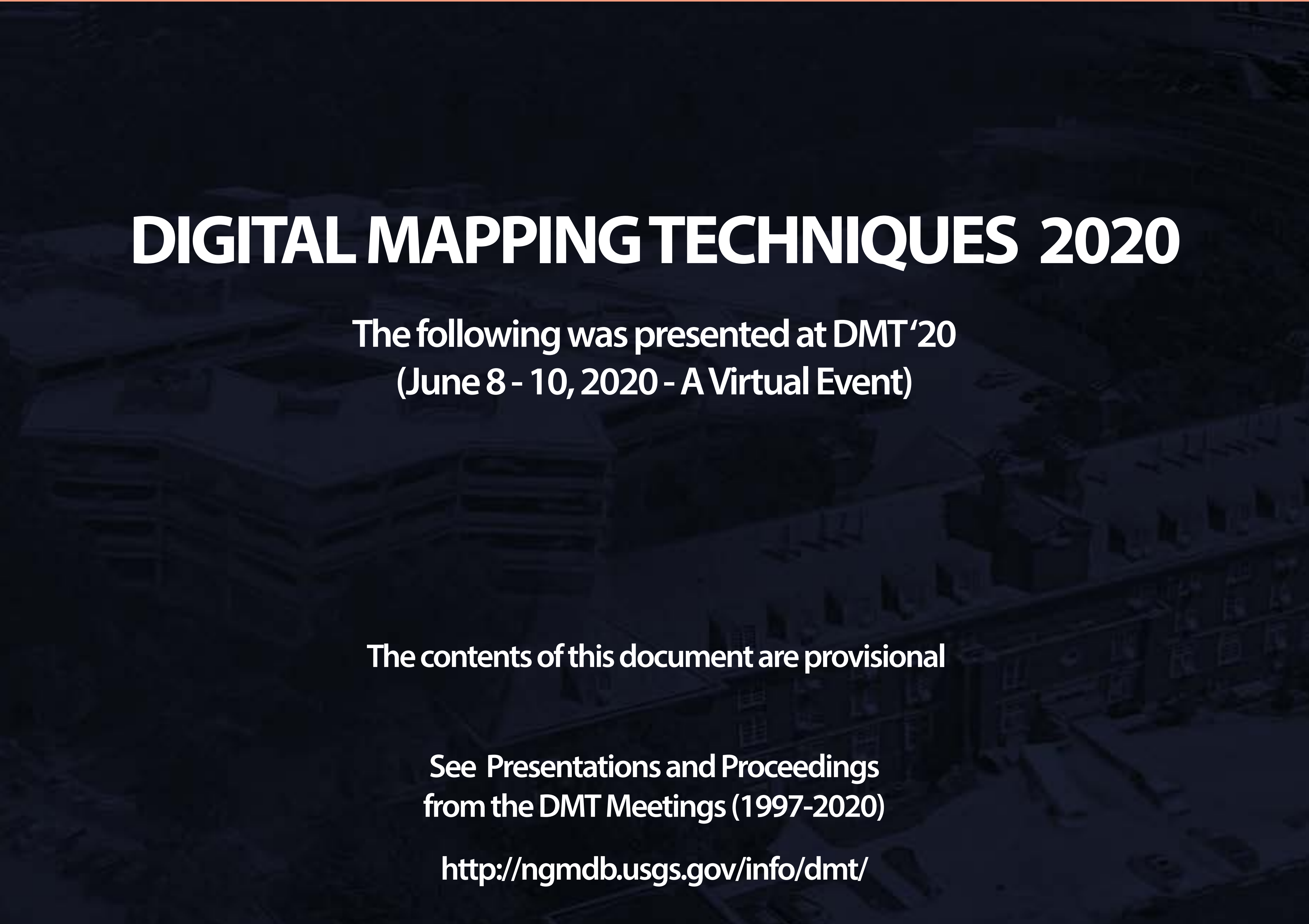


DIGITAL MAPPING TECHNIQUES 2020

Attribute Rules and Dictionary Symbology in ArcGIS Pro Help Streamline Geologic Map Compilation in GeMS

Tracey J. Felger
USGS, Geology, Minerals, Energy, and Geophysics Science Center
Flagstaff, AZ
(tfelger@usgs.gov)

Talk Overview

- **Background**
- **What are Attribute Rules?**
  - Test Case – apply to ‘Traditional’ workflow
- **What is Dictionary Symbology?**
  - Test Case – apply to ‘Granular’ data-driven workflow
- **Combining Attribute Rules and Dictionary Symbology for an Optimized Workflow**
- **Summary**

Background

- **Current projects -** *all funded by the National Cooperative Geologic Mapping Program (NCGMP)*
  - Mojave Desert/Eastern California Shear Zone (MOJO)
  - Lower Colorado River (LOCOS)
  - Enterprise GIS for the NCGMP Community
- **Goals for Fiscal Year 2020**
  - Embrace the *Ge*ologic *M*ap *S*chema (GeMS), which is mandated by NCGMP, instead of just coping with it!
  - Transition compilation workflow from ArcMap to ArcPro in order to leverage new functionality that would improve compilation efficiency and data quality

Traditional vs Granular Approach

- **Traditional (Cartography-driven) –**the symbol and the data are intertwined, and can be easily represented in one field (e.g. ALACARTE)
  - Main purpose is to make a cartographic product
- **Granular (Data-driven)** – information is disaggregated into the smallest pieces possible, and is stored in many fields (e.g. GeMS)

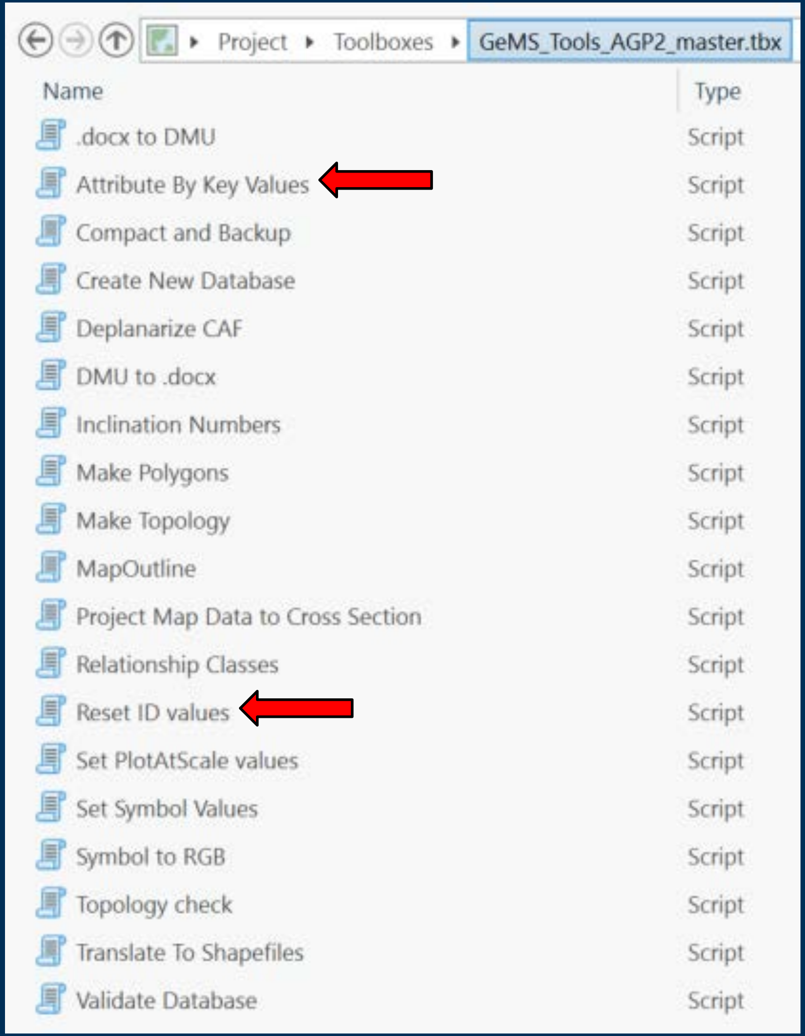
Feature Layer	ALACARTE Fields	GeMS Fields
Contacts and Faults	1	9
Map Units	1	6
Structural Measurements	3	12

- Machine readable (good for analysis)
- Can be aggregated and disaggregated in different ways as needed
- Easily merged with other data and effectively integrated and managed
- *Better for making a multi-map compilation!*

Coping Strategy…

- **Populate 1 or 2 fields while editing and then use scripts in the GeMS toolbox, such as Attribute By Key Values and Reset ID Values (shown in snapshot to right) to populate other fields, either when map is finished, or periodically during compilation**
  - GeMS toolbox (https://github.com/usgs/gems-tools-pro)
  - Ryan Crow's Map Extractor and Schema Converter tools includes other useful scripts not available in the GeMS toolbox (https://github.com/rcrow/MapExtractor_SchemaConverter)

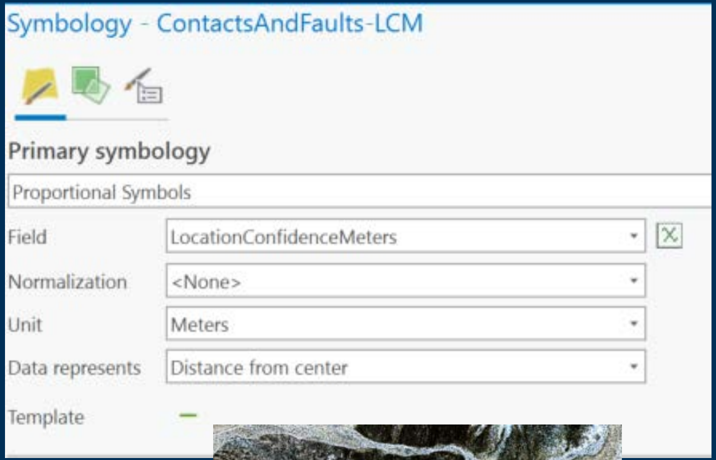
This works, but for me it isn't very satisfying, and is an obstacle to embracing GeMS!



Name	Type
.docx to DMU	Script
Attribute By Key Values	Script
Compact and Backup	Script
Create New Database	Script
Deplanarize CAF	Script
DMU to .docx	Script
Inclination Numbers	Script
Make Polygons	Script
Make Topology	Script
MapOutline	Script
Project Map Data to Cross Section	Script
Relationship Classes	Script
Reset ID values	Script
Set PlotAtScale values	Script
Set Symbol Values	Script
Symbol to RGB	Script
Topology check	Script
Translate To Shapefiles	Script
Validate Database	Script

December, 2019 - Started compiling new MOJO mapping

- Downloaded the GeMS toolbox for Pro (https://github.com/usgs/gems-tools-pro)
- Ran the Create New Database script (and it worked – yay!) Create New Database
  - *It took 12 minutes to create an empty gdb with all the bells and whistles – the ArcMap version took < 2 minutes!*
- Symbolized ContactsAndFaults twice, once on Symbol, and a second time on LocationConfidenceMeters (LCM)
  - *This is a neat trick that Ralph demonstrated at the 2019 DMT that makes it very easy to visualize LCM spatially*
- Started compiling using a 'traditional' workflow
  - Digitize ContactsAndFaults - Populate Symbol, LCM, and DataSourceID fields
  - Digitize MapUnitPoints - Populate MapUnit and DataSourceID fields
  - Build MapUnitPolys from ContactsAndFaults and MapUnitPoints
- Ran the Validate Database script (and it worked – yay!) Validate Database
  - *But I had a lot of errors because of unpopulated fields (boo-hiss!)*
- Run Reset ID values, Attribute by Key Values; rerun Validate Database (*sigh…*) Reset ID values | Attribute By Key Values | Validate Database


Symbology - ContactsAndFaults-LCM

Primary symbology: Proportional Symbols

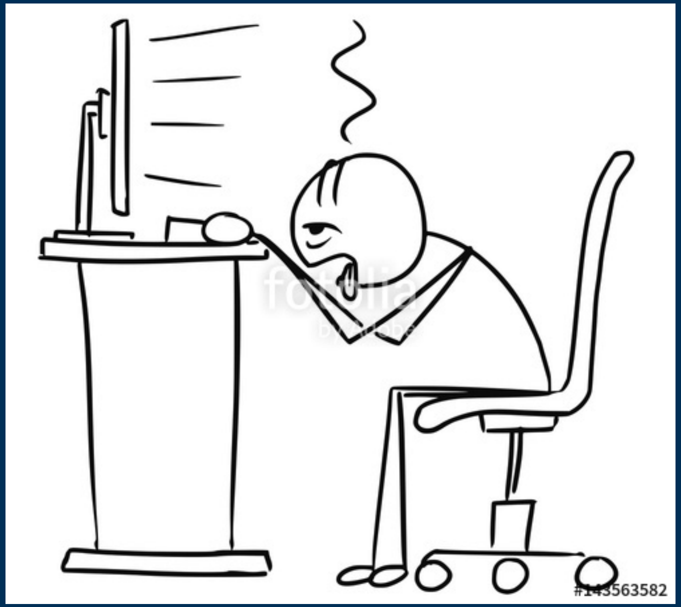
Field: LocationConfidenceMeters

Normalization: <None>

Unit: Meters

Data represents: Distance from center

Template



Tracey

Can Attribute Rules Lighten the GeMS Workload?

- **Attribute rules:**
  - user-defined Arcade scripts embedded within a geodatabase

    **Arcade is a lightweight scripting language specific to the ESRI platform; similar to JavaScript*
  - triggered automatically by editing operations or can be run on-demand

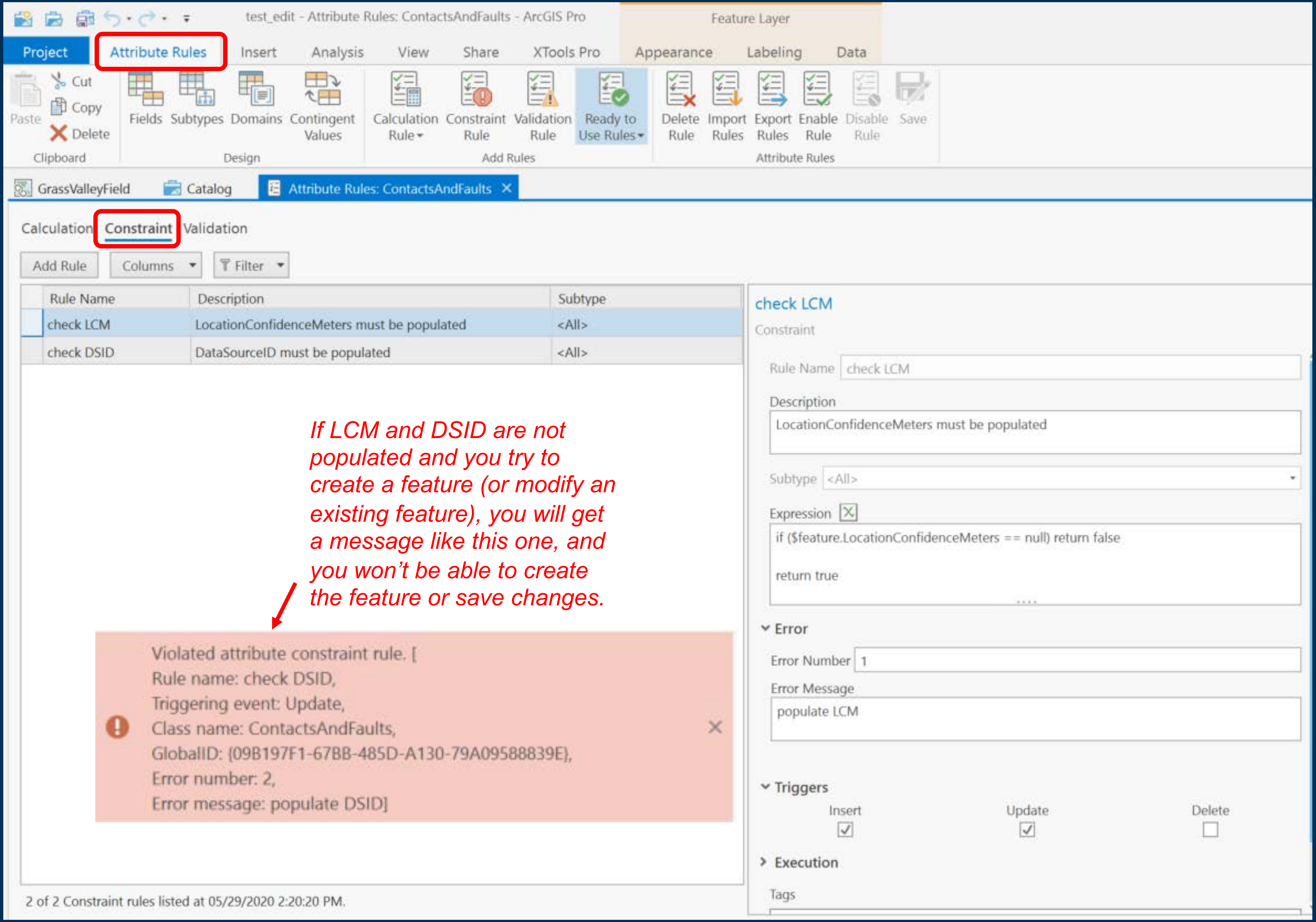
    **Similar to formulas and macros in Microsoft Excel, and triggers and stored procedures in SQL*
  - 3 types of rules:
    - calculation rules - automatically populate attributes
    - constraint rules - restrict invalid edits during edit operations
    - validation rules - perform quality assurance checks on existing features

Test Case – Apply Attribute Rules to a 'Traditional' Workflow

- **Goals:**
  - Don't let me create or modify features unless certain fields are populated
  - Automatically populate feature_ID field
    - *so I don't need to run the Reset ID values script*
  - Automatically populate dependent fields based on the value in Symbol
    - *so I don't need to run the Attribute By Key Values script*
- **I guess I need to learn Arcade!**

  **Arcade is a lightweight scripting language specific to the ESRI platform; similar to JavaScript*

ContactsAndFaults (Traditional) – Constraint Rules



Rule Name	Description	Subtype
check LCM	LocationConfidenceMeters must be populated	<All>
check DSID	DataSourceID must be populated	<All>

If LCM and DSID are not populated and you try to create a feature (or modify an existing feature), you will get a message like this one, and you won't be able to create the feature or save changes.

```
Violated attribute constraint rule. [
Rule name: check DSID,
Triggering event: Update,
Class name: ContactsAndFaults,
GlobalID: {09B197F1-67BB-485D-A130-79A09588839E},
Error number: 2,
Error message: populate DSID]
```

2 of 2 Constraint rules listed at 05/29/2020 2:20:20 PM.

check LCM

Constraint

Rule Name: check LCM

Description: LocationConfidenceMeters must be populated

Subtype: <All>

Expression:

```
if ($feature.LocationConfidenceMeters == null) return false

return true
```

Error

Error Number: 1

Error Message: populate LCM

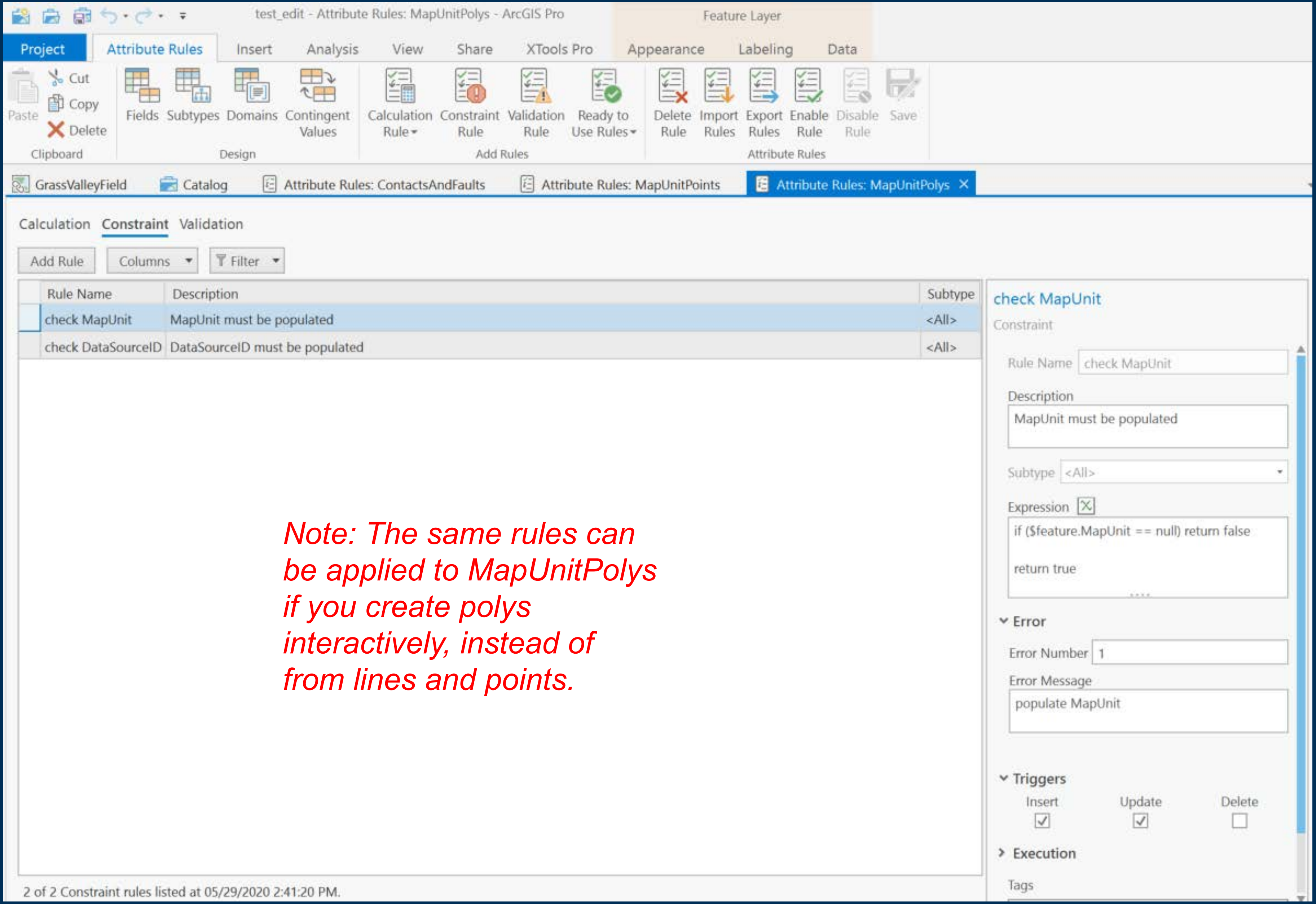
Triggers

Insert ✓ Update ✓ Delete ☐

Execution

Tags

MapUnit Points (Traditional) – Constraint Rules



Rule Name	Description	Subtype
check MapUnit	MapUnit must be populated	<All>
check DataSourceID	DataSourceID must be populated	<All>

2 of 2 Constraint rules listed at 05/29/2020 2:41:20 PM.

check MapUnit
Constraint

Rule Name: check MapUnit

Description: MapUnit must be populated

Subtype: <All>

Expression:
```
if ($feature.MapUnit == null) return false

return true
```

Error

Error Number: 1

Error Message: populate MapUnit

Triggers

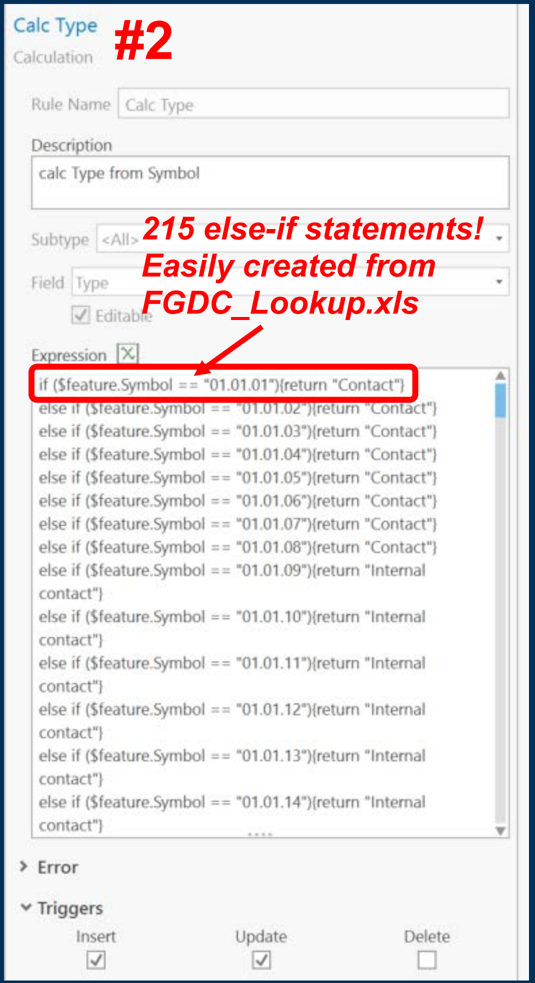
Insert ✓ Update ✓ Delete ☐

Execution

Tags

Note: The same rules can be applied to MapUnitPolys if you create polys interactively, instead of from lines and points.

ContactsAndFaults (Traditional) – Calculation Rules



#2

Calc Type
Calculation

Rule Name: Calc Type

Description: calc Type from Symbol

Subtype: <All>

Field: Type

✓ Editable

215 else-if statements! Easily created from FGDC_Lookup.xls

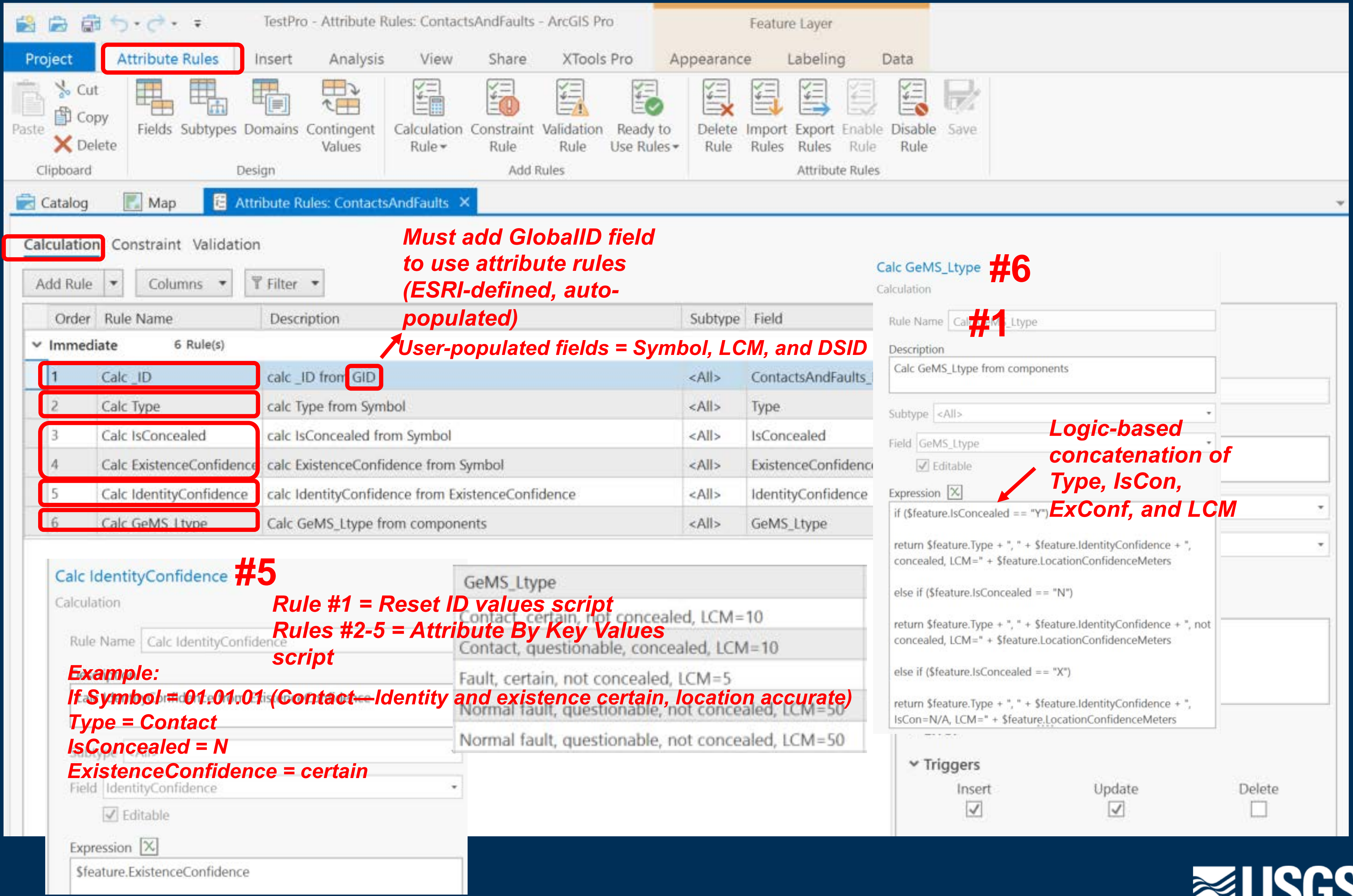
Expression

```
if ($feature.Symbol == "01.01.01"){return "Contact"}
else if ($feature.Symbol == "01.01.02"){return "Contact"}
else if ($feature.Symbol == "01.01.03"){return "Contact"}
else if ($feature.Symbol == "01.01.04"){return "Contact"}
else if ($feature.Symbol == "01.01.05"){return "Contact"}
else if ($feature.Symbol == "01.01.06"){return "Contact"}
else if ($feature.Symbol == "01.01.07"){return "Contact"}
else if ($feature.Symbol == "01.01.08"){return "Contact"}
else if ($feature.Symbol == "01.01.09"){return "Internal
contact"}
else if ($feature.Symbol == "01.01.10"){return "Internal
contact"}
else if ($feature.Symbol == "01.01.11"){return "Internal
contact"}
else if ($feature.Symbol == "01.01.12"){return "Internal
contact"}
else if ($feature.Symbol == "01.01.13"){return "Internal
contact"}
else if ($feature.Symbol == "01.01.14"){return "Internal
contact"}
```

Error

Triggers

Insert	Update	Delete
✓	✓	☐



Must add GlobalID field to use attribute rules (ESRI-defined, auto-populated)

User-populated fields = Symbol, LCM, and DSID

Immediate 6 Rule(s)

Order	Rule Name	Description	Subtype	Field
1	Calc _ID	calc _ID from GID	<All>	ContactsAndFaults_
2	Calc Type	calc Type from Symbol	<All>	Type
3	Calc IsConcealed	calc IsConcealed from Symbol	<All>	IsConcealed
4	Calc ExistenceConfidence	calc ExistenceConfidence from Symbol	<All>	ExistenceConfidence
5	Calc IdentityConfidence	calc IdentityConfidence from ExistenceConfidence	<All>	IdentityConfidence
6	Calc GeMS_Ltype	Calc GeMS_Ltype from components	<All>	GeMS_Ltype

#5

Calc IdentityConfidence
Calculation

Rule Name: Calc IdentityConfidence

Field: IdentityConfidence

✓ Editable

Expression

```
$feature.ExistenceConfidence
```

GeMS_Ltype
Contact, certain, not concealed, LCM=10
Contact, questionable, concealed, LCM=10
Fault, certain, not concealed, LCM=5
Normal fault, questionable, not concealed, LCM=50
Normal fault, questionable, not concealed, LCM=50

Rule #1 = Reset ID values script
Rules #2-5 = Attribute By Key Values script

Example:
If Symbol = 01.01.01 (Contact—Identity and existence certain, location accurate)
Type = Contact
IsConcealed = N
ExistenceConfidence = certain

#6 **#1**

Calc GeMS_Ltype
Calculation

Rule Name: Calc GeMS_Ltype

Description: Calc GeMS_Ltype from components

Subtype: <All>

Field: GeMS_Ltype

✓ Editable

Logic-based concatenation of Type, IsCon, ExConf, and LCM

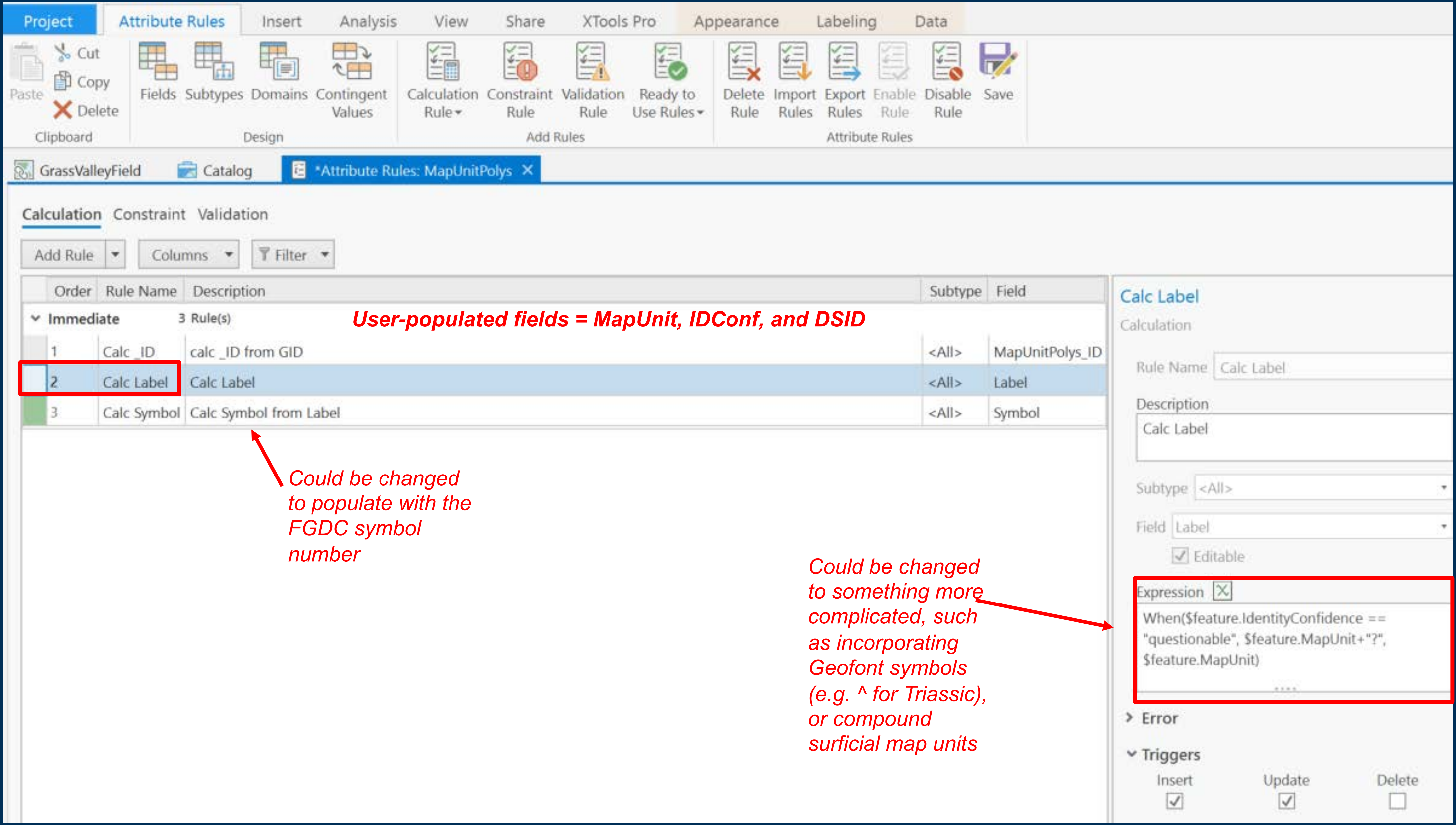
Expression

```
if ($feature.IsConcealed == "Y")

return $feature.Type + ", " + $feature.IdentityConfidence + ",
concealed, LCM=" + $feature.LocationConfidenceMeters

else if ($feature.IsConcealed == "N")

return $feature.Type + ", " + $feature.IdentityConfidence + ", not
concealed, LCM=" + $feature.LocationConfidenceMeters

else if ($feature.IsConcealed == "X")

return $feature.Type + ", " + $feature.IdentityConfidence + ",
IsCon=N/A, LCM=" + $feature.LocationConfidenceMeters
```

Triggers

Insert	Update	Delete
✓	✓	☐

MapUnitPoints (Traditional) – Calculation Rules



Order	Rule Name	Description	Subtype	Field
Immediate	3 Rule(s)			
1	Calc _ID	calc _ID from GID	<All>	MapUnitPolys_ID
2	Calc Label	Calc Label	<All>	Label
3	Calc Symbol	Calc Symbol from Label	<All>	Symbol

User-populated fields = MapUnit, IDConf, and DSID

Could be changed to populate with the FGDC symbol number

Calc Label
Calculation

Rule Name: Calc Label
Description: Calc Label
Subtype: <All>
Field: Label
✓ Editable

Expression

```
When($feature.IdentityConfidence ==
"questionable", $feature.MapUnit+"?",
$feature.MapUnit)
```

Could be changed to something more complicated, such as incorporating Geofont symbols (e.g. ^ for Triassic), or compound surficial map units

Error
Triggers

Insert	Update	Delete
✓	✓	☐

Attribute Rules –Test Case Results

Yay! I don't have to run Reset ID Values and Attribute By Key Values anymore!

Goals:

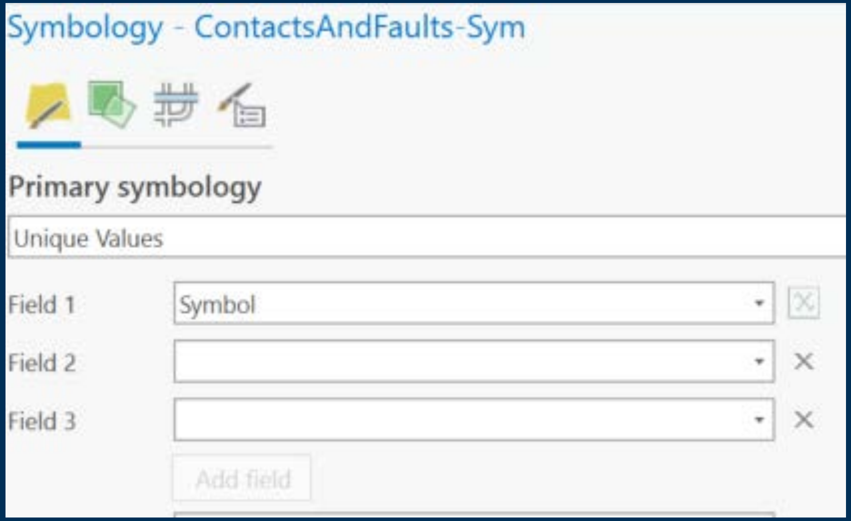
- ✓ Don't let me create or modify features unless certain fields are populated
- ✓ Automatically populate _ID
- ✓ Automatically populate dependent fields based on the value in Symbol

Tracey

But I'm still thinking in a cartographic, not 'granular', data-driven way, partly because of the way we symbolize our data…

ContactsAndFaults (Traditional) Symbology

- **Symbolize on unique values in a field**
  - E.g. Match to FGDC_GSC.style on Symbol
  - All symbol permutations must exist
  - Limited to unique values in no more than 3 fields


Symbology - ContactsAndFaults-Sym

Primary symbology: Unique Values

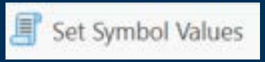
Field 1: Symbol

Field 2:

Field 3:

But the intent of GeMS is that the symbol would be calculated on the basis of map scale and the attributes Type, IsConcealed, LocationConfidenceMeters, ExistenceConfidence, and IdentityConfidence

- *The Set Symbol Values script will do this, but I want to symbolize on all of those attributes while I'm editing!*

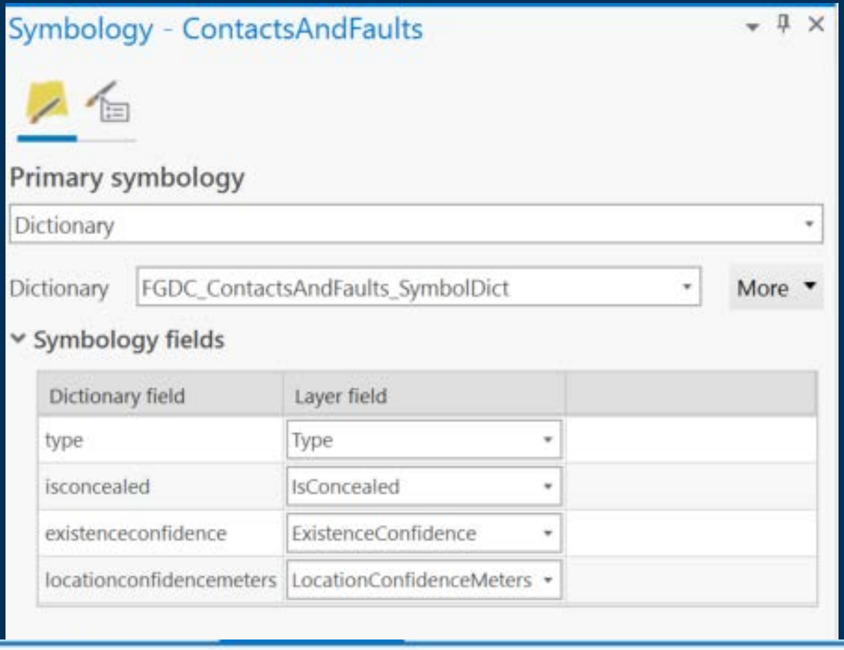


Can Dictionary Symbology Help?

- **Dictionary Symbology:**
  - An ‘intelligent’ style file (mobile .stylx) stored in an SQLite database that contains:
    - Symbol ‘building blocks’
    - Embedded configuration parameters (JSON)
      - Specifies which fields are used for symbology, which fields are used for text, and properties that can be user-controlled
    - Embedded dictionary of rules
      - user-defined Arcade scripts that control how symbol components are aggregated to visually represent the data, depending on the feature attributes
  - Builds symbols on-the-fly from components, based on one or more attributes of a feature, and logic (optional)

Very powerful and complicated - hard to explain in < 20 minutes!

Dictionary Symbology – Interface and functionality



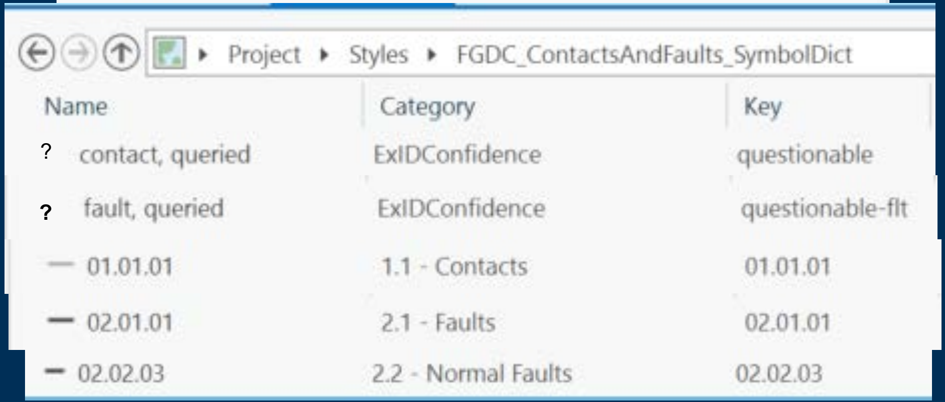
Symbology - ContactsAndFaults

Primary symbology: Dictionary

Dictionary: FGDC_ContactsAndFaults_SymbolDict

Symbology fields

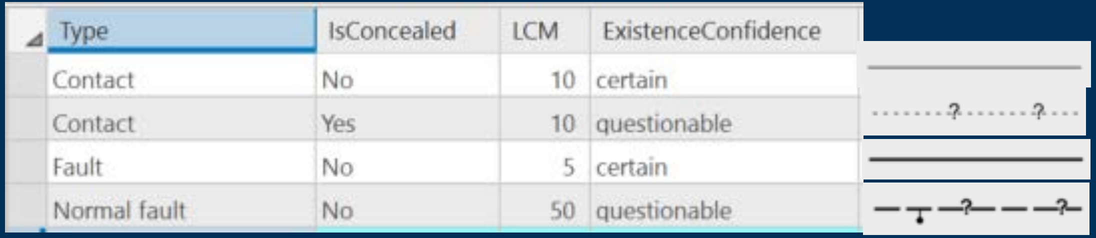
Dictionary field	Layer field
type	Type
isconcealed	IsConcealed
existenceconfidence	ExistenceConfidence
locationconfidencemeters	LocationConfidenceMeters



Project ▸ Styles ▸ FGDC_ContactsAndFaults_SymbolDict

Name	Category	Key
? contact, queried	ExIDConfidence	questionable
? fault, queried	ExIDConfidence	questionable-flt
— 01.01.01	1.1 - Contacts	01.01.01
— 02.01.01	2.1 - Faults	02.01.01
— 02.02.03	2.2 - Normal Faults	02.02.03

1. **Get the value in Type (e.g. Contact)**
2. **Get the value in IsConcealed (IsCon)**
   - If IsCon = Y, get a dotted contact line
   - If IsCon = N, go to Step #3
3. **Get the value in LocationConfidenceMeters (LCM)**
   - If LCM <= 20, get a solid contact line
   - If LCM >20 and <50, get a short-dash contact line
   - If LCM >=50, get a long-dash contact line
4. **Get the value in ExistenceConfidence (ExConf)**
   - If ExConf = questionable, get a contact-weight questionmark
5. **Symbolize the line with the symbol parts ‘collected’ in steps 1-4**



Type	IsConcealed	LCM	ExistenceConfidence
Contact	No	10	certain
Contact	Yes	10	questionable
Fault	No	5	certain
Normal fault	No	50	questionable

Dictionary Symbology –Test Case Results

Yay! Now I can interact with my data on a granular level!

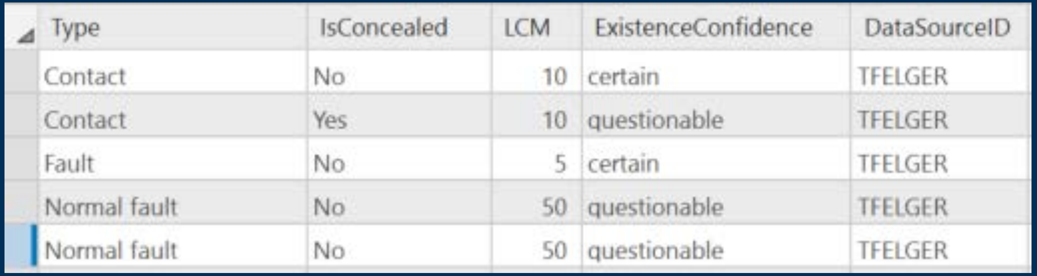
Tracey

USGS

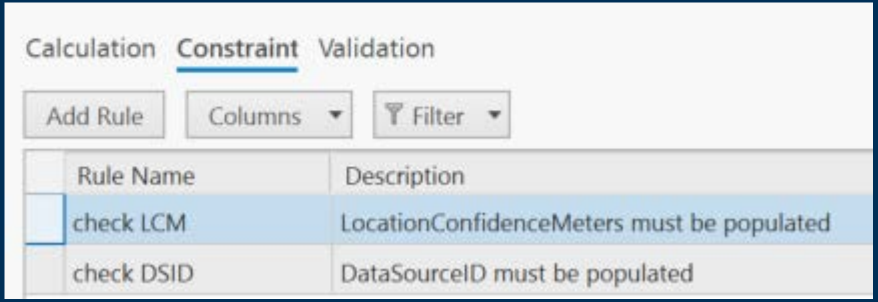
Attribute Rules + Dictionary Symbology = Optimized GeMS Workflow

ContactsAndFaults – Optimized Workflow

#1. Populate Type, IsCon, LCM, ExConf, and DSID, and use Constraint rules to make sure LCM and DSID are populated

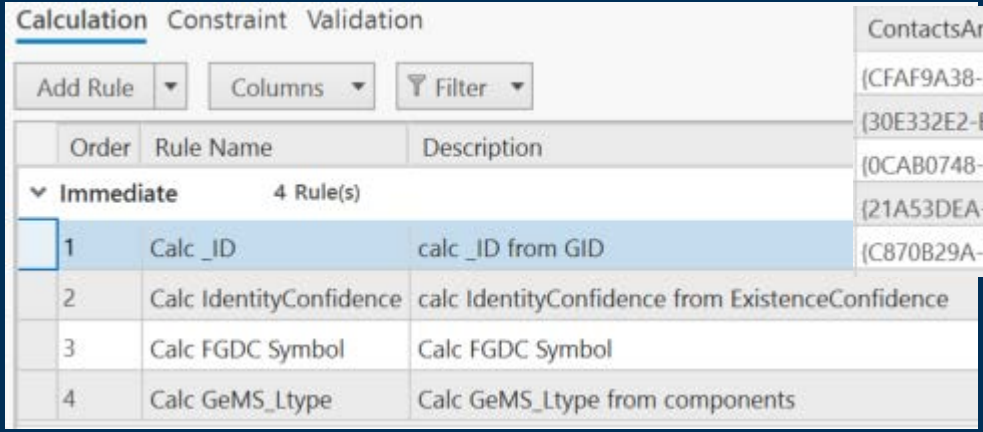


Type	IsConcealed	LCM	ExistenceConfidence	DataSourceID
Contact	No	10	certain	TFELGER
Contact	Yes	10	questionable	TFELGER
Fault	No	5	certain	TFELGER
Normal fault	No	50	questionable	TFELGER
Normal fault	No	50	questionable	TFELGER

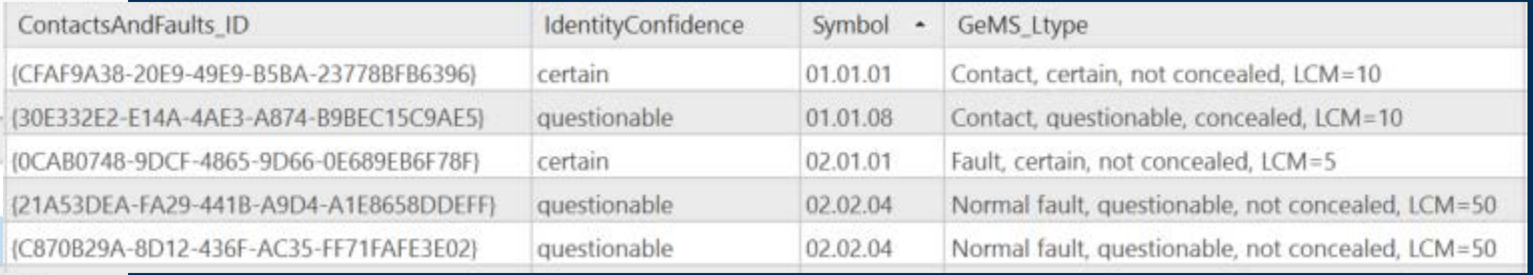


Rule Name	Description
check LCM	LocationConfidenceMeters must be populated
check DSID	DataSourceID must be populated

#2. Use Calculation rules to populate _ID, IdentityConfidence, Symbol, and GeMS_Ltype

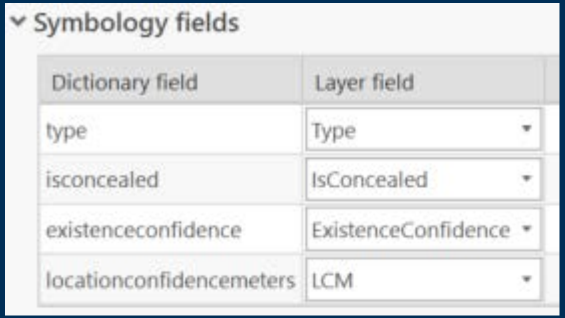


Order	Rule Name	Description
Immediate	4 Rule(s)	
1	Calc _ID	calc _ID from GID
2	Calc IdentityConfidence	calc IdentityConfidence from ExistenceConfidence
3	Calc FGDC Symbol	Calc FGDC Symbol
4	Calc GeMS_Ltype	Calc GeMS_Ltype from components



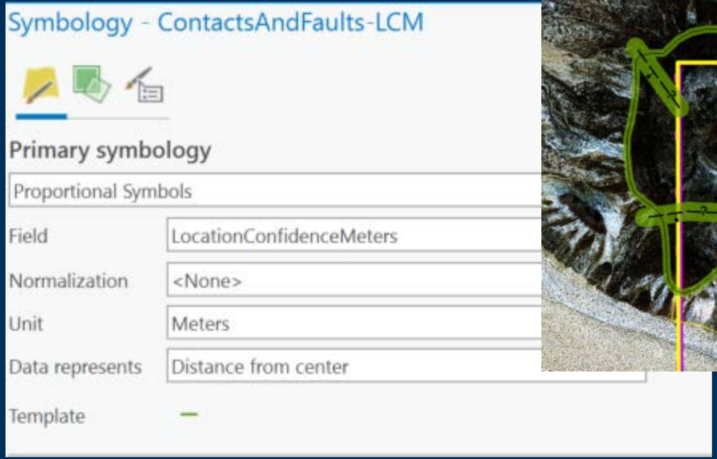
ContactsAndFaults_ID	IdentityConfidence	Symbol	GeMS_Ltype
{CFAF9A38-20E9-49E9-B5BA-23778BFB6396}	certain	01.01.01	Contact, certain, not concealed, LCM=10
{30E332E2-E14A-4AE3-A874-B9BEC15C9AE5}	questionable	01.01.08	Contact, questionable, concealed, LCM=10
{0CAB0748-9DCF-4865-9D66-0E689EB6F78F}	certain	02.01.01	Fault, certain, not concealed, LCM=5
{21A53DEA-FA29-441B-A9D4-A1E8658DDEFF}	questionable	02.02.04	Normal fault, questionable, not concealed, LCM=50
{C870B29A-8D12-436F-AC35-FF71FAFE3E02}	questionable	02.02.04	Normal fault, questionable, not concealed, LCM=50

#3. Symbolize twice – once using Dictionary Symbology, again on LocationConfidenceMeters



Symbology fields

Dictionary field	Layer field
type	Type
isconcealed	IsConcealed
existenceconfidence	ExistenceConfidence
locationconfidencemeters	LCM



Symbology - ContactsAndFaults-LCM

Primary symbology

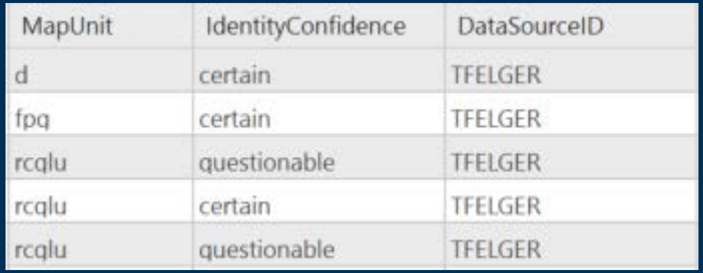
Proportional Symbols

Field	LocationConfidenceMeters
Normalization	<None>
Unit	Meters
Data represents	Distance from center
Template	—

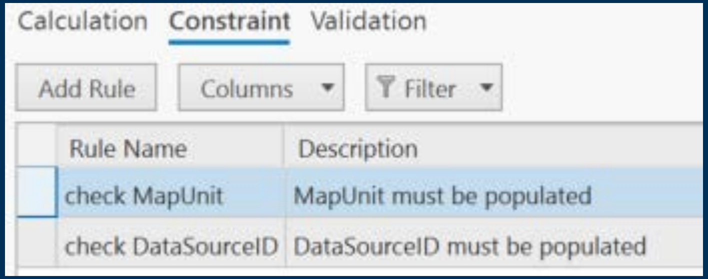
USGS

MapUnitPolys – Optimized Workflow

#1. Create polys interactively, and populate MapUnit, IdConf, and DSID, and use Constraint rules to make sure MapUnit and DSID are populated

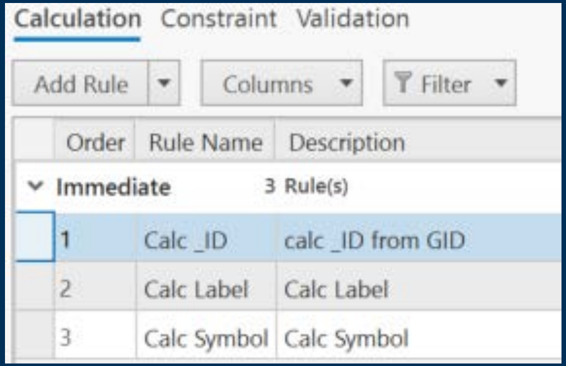


MapUnit	IdentityConfidence	DataSourceID
d	certain	TFELGER
fpq	certain	TFELGER
rcqlu	questionable	TFELGER
rcqlu	certain	TFELGER
rcqlu	questionable	TFELGER

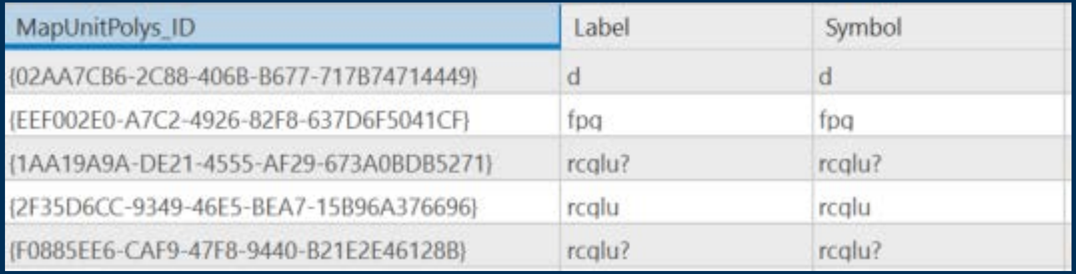


Rule Name	Description
check MapUnit	MapUnit must be populated
check DataSourceID	DataSourceID must be populated

#2. Use Calculation rules to populate _ID, Label, and Symbol

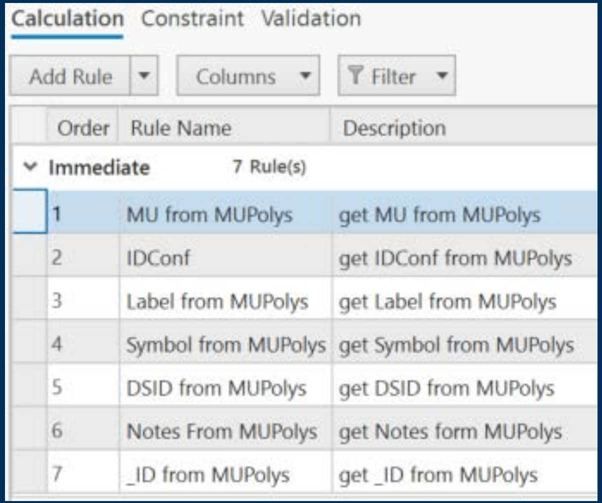


Order	Rule Name	Description
Immediate	3 Rule(s)	
1	Calc _ID	calc _ID from GID
2	Calc Label	Calc Label
3	Calc Symbol	Calc Symbol

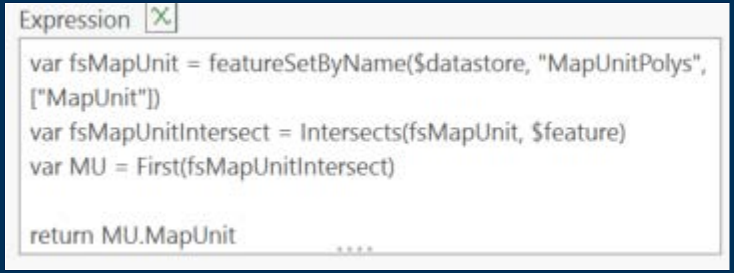


MapUnitPolys_ID	Label	Symbol
{02AA7CB6-2C88-406B-B677-717B74714449}	d	d
{EEF002E0-A7C2-4926-82F8-637D6F5041CF}	fpq	fpq
{1AA19A9A-DE21-4555-AF29-673A0BDB5271}	rcqlu?	rcqlu?
{2F35D6CC-9349-46E5-BEA7-15B96A376696}	rcqlu	rcqlu
{F0885EE6-CAF9-47F8-9440-B21E2E46128B}	rcqlu?	rcqlu?

#3. Create polys interactively from ContactsAndFaults, but maintain a MapUnitPoints layer in case I don't stick with that! Use Calculation rules to get all attributes from 'parent' poly



Order	Rule Name	Description
Immediate	7 Rule(s)	
1	MU from MUPolys	get MU from MUPolys
2	IDConf	get IDConf from MUPolys
3	Label from MUPolys	get Label from MUPolys
4	Symbol from MUPolys	get Symbol from MUPolys
5	DSID from MUPolys	get DSID from MUPolys
6	Notes From MUPolys	get Notes form MUPolys
7	_ID from MUPolys	get _ID from MUPolys



```
var fsMapUnit = featureSetByName($datastore, "MapUnitPolys",
["MapUnit"])
var fsMapUnitIntersect = Intersects(fsMapUnit, $feature)
var MU = First(fsMapUnitIntersect)

return MU.MapUnit
```

Summary *Game changer!*

Benefits:

1. **Improved efficiency** – attributes automatically populated on the fly, no need to periodically run several scripts
2. **Improved data quality** –
   a) features can't be created/modified unless certain fields are populated
   b) all fields are always fully populated, so Validate database script produces fewer errors
3. **Helping me embrace GeMS instead of just coping with it!**

Challenges and Limitations:

1. Can only use in Pro - not backward compatible with ArcMap *(there are workarounds)*
2. Have to learn Arcade
3. Dictionary symbology is very powerful, but very complicated and not well documented!!

Testing so far:

- Spatial data only (haven't tested on non-spatial tables)
- ContactsAndFaults and MapUnitPts/Polys only
- Calculation and constraint rules only (haven't tried validation rules) *(I'm happy to share!)*
- Limited to 215 FGDC contacts and fault types so far
- Latitude 7212 ruggedized tablet; Windows 10, Pro 2.5 (I started in 2.4; after upgrading to 2.5 the GeMS Validate Database script stopped working!)

Next Steps:

- Expand and refine as I need to for my mapping
- Try validation rules
- Dictionary symbology - implement user-control options, and symbol override capabilities