

データデバッグツールの重要性

株式会社セガ クリエイティブセンター
緒方 修

今日のお話

1：現状認識

2：問題点

3：どうすればいい？

4：データデバッグツール

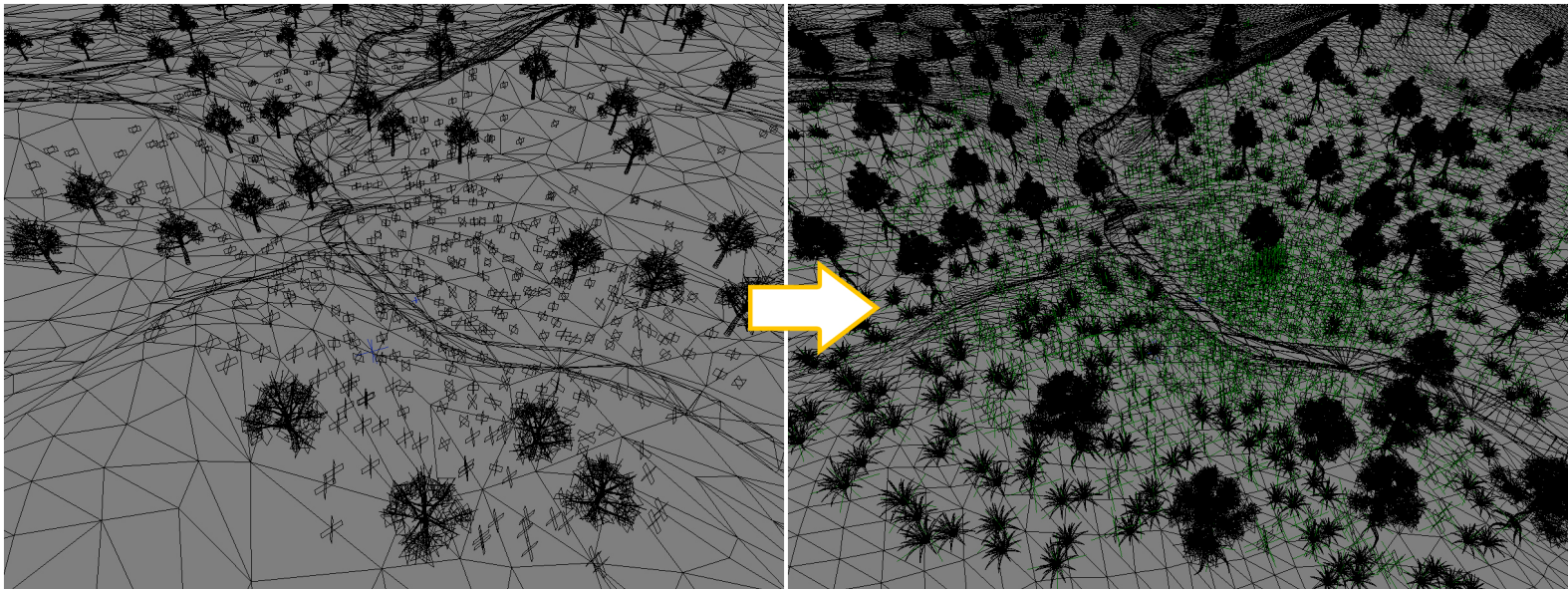
5：落とし穴

6：本当はその前に

何故データデバッグツールが重要なのか？
我々を取り巻く現状をおさらいしてみよう。

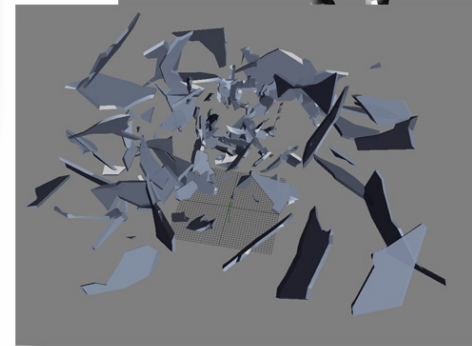
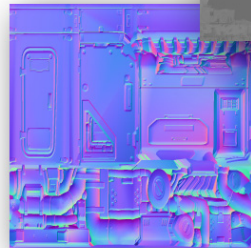
現状認識

- 作る量が増えた。



現状認識

- データの種類も増えた。



現状認識

- ルールは複雑化した。

```
Float4 B = N.yz * I.yz;
B = (-I.yz * N.z + N.z * I.yz) + B;

// @ワールド空間に変換した法線、接線、法線を出力*/
OUT.n = mul(World[1], I.yz);
OUT.t = mul(World[1], I.yz);
OUT.b = mul(World[1], I.yz);

// @ノーマルベクトルをワールド空間へ*/
Float4 Po = Float4(N.yz, 1);
Float3 wpos = mul(Wor[16], Po);

// @このデータを使う。ワールド空間へ*/
Float3 wvert = mul(Wor[16], wpos);
// @それぞれのデータをフラグメントに渡す*/
OUT.n = wvert;
OUT.t = wpos;
OUT.b = mul(Wor[16], wpos);
return OUT;

// 頂点シェーダ終わり出力*/
// フラグメント (ピクセル) シェーダ始まり出力*/
Float4 ResPixelShader(vertexOutput IN) : COLOR;

// @ワールド空間に変換した法線、接線、法線をINして正規化*/
Float3 n = normalize(IN.n);
Float3 t = normalize(IN.t);
Float3 b = normalize(IN.b);

// @の法線マップを生成して読み込む*/
Float3 Norm = normalize(tex2D(NormaMap, IN.tc0.yz - 0.5));
Norm.y = 1 - Norm.y; Norm.z = EnvScale;
Float3 m = Norm.x + v1.yz + Norm.y * v2.yz + Norm.z * v3.yz;
m = normalize(m);

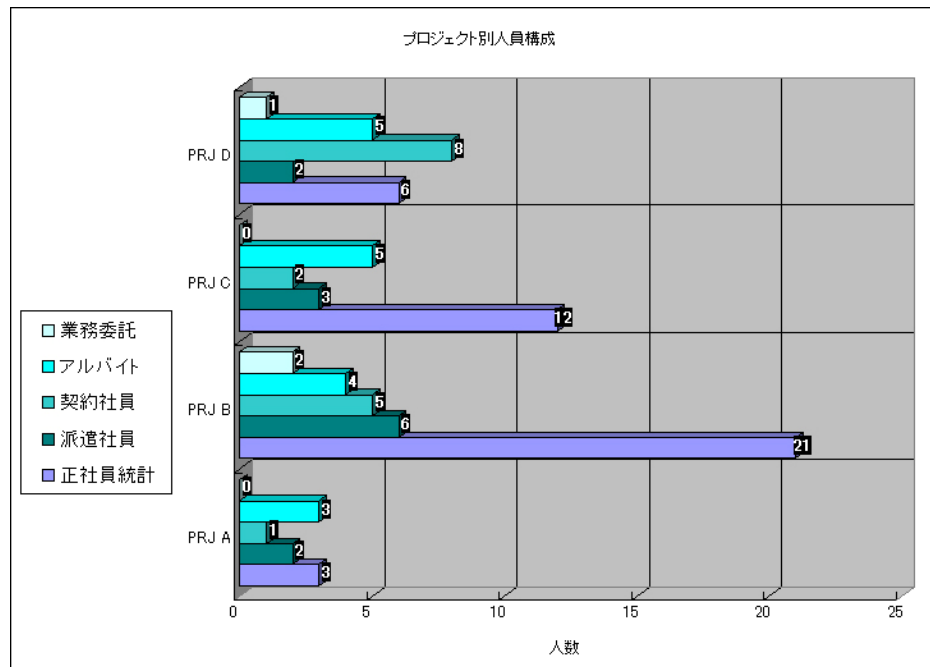
// @反射ベクトル作成*/
Float3 R = reflect(-IN.v, m);
R = normalize(R);

// @鏡射的環境マップ用の反射ベクトルに変換*/
Float3 RI = IN.n * R;
Float3 RI = 1 + (sqrt(dot(T, T)) + pow(EnvScale, 2)) - abs(T) + R;

// @ニューアップ環境マップを計算*/
Float4 texture = tex2D(EnvMap, RI);
// @環境マップを生成*/
```

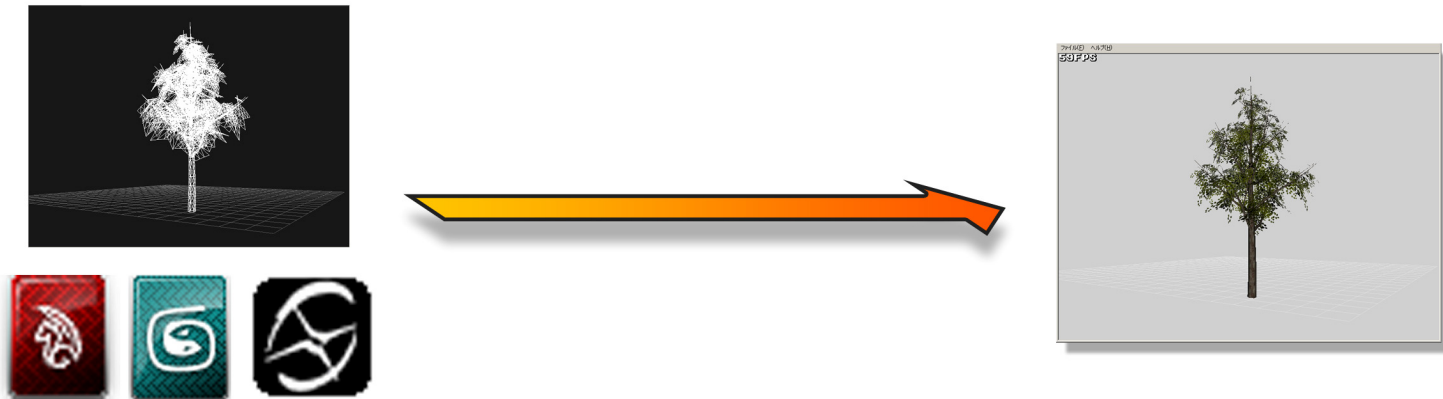
現状認識

- 社員だけでチームを構成するのは難しくなった。



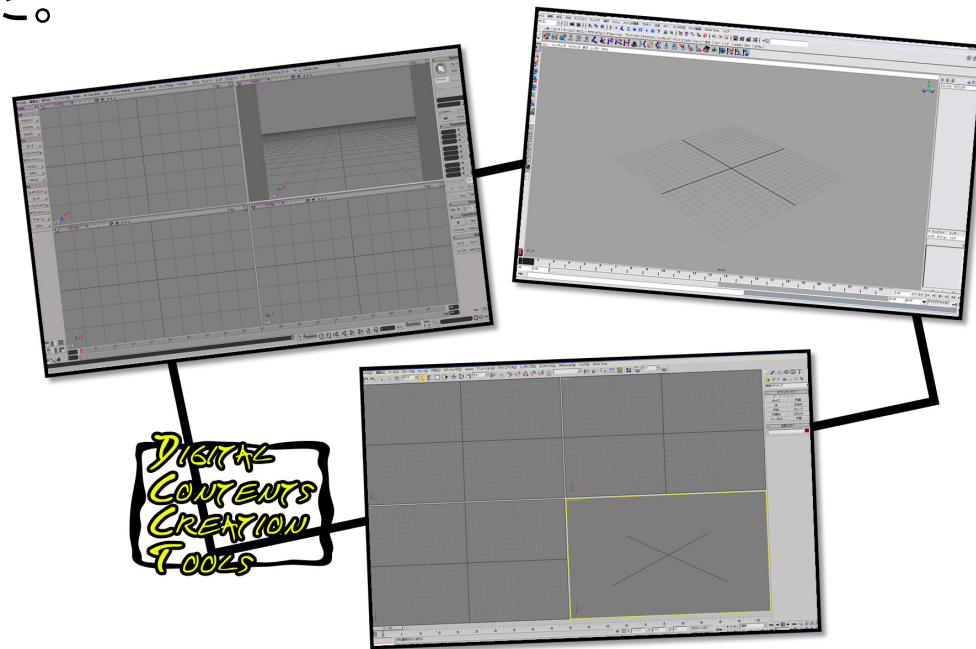
現状認識

- データフローはそれほど変化していない。



現状認識

- 既存のDCCツールも変化していない。
※「作る」機能は増えた。

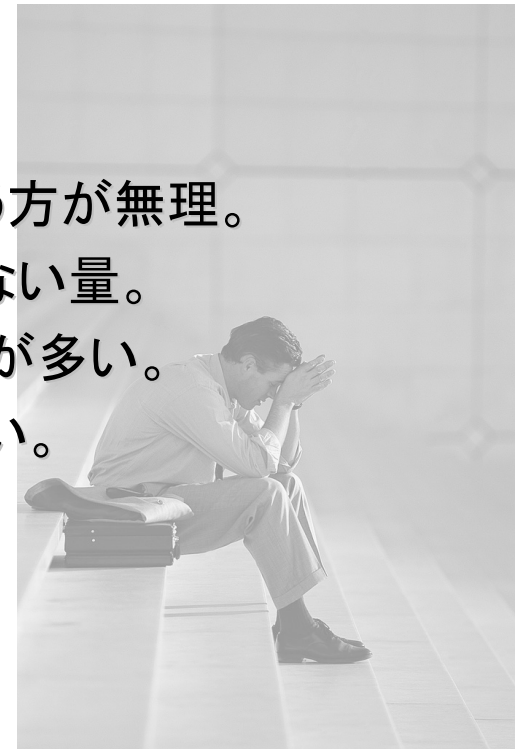


それらが複合して、以前より遥かにデータ
内容の確認は難しくなった。
それに伴って問題が発生するようになった。

問題点

- 検収作業が多くなった。

量が多くて、ルールが複雑なのに間違ふなという方が無理。
外注からあがって来るデータ検収もバカにならない量。
それはラインチーフデザイナーの仕事である場合が多い。
しかし、検収作業「だけ」なんて、誰もやりたくない。
一生それかと思うと辞めたくなる。

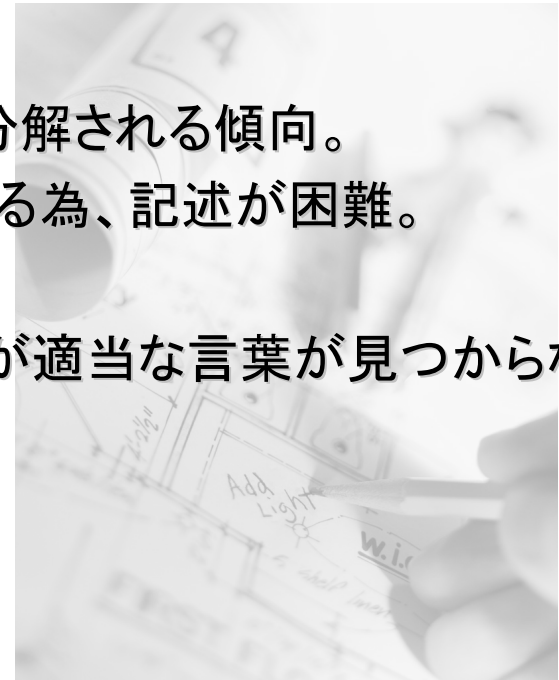


問題点

- 仕様書に条件を書きにくくなった。

人数が増え、分業化も進み、ルールも細かく分解される傾向。
しかし、そのルールは非常に難解で多岐に渡る為、記述が困難。
条件が分岐する場合も多々ある。

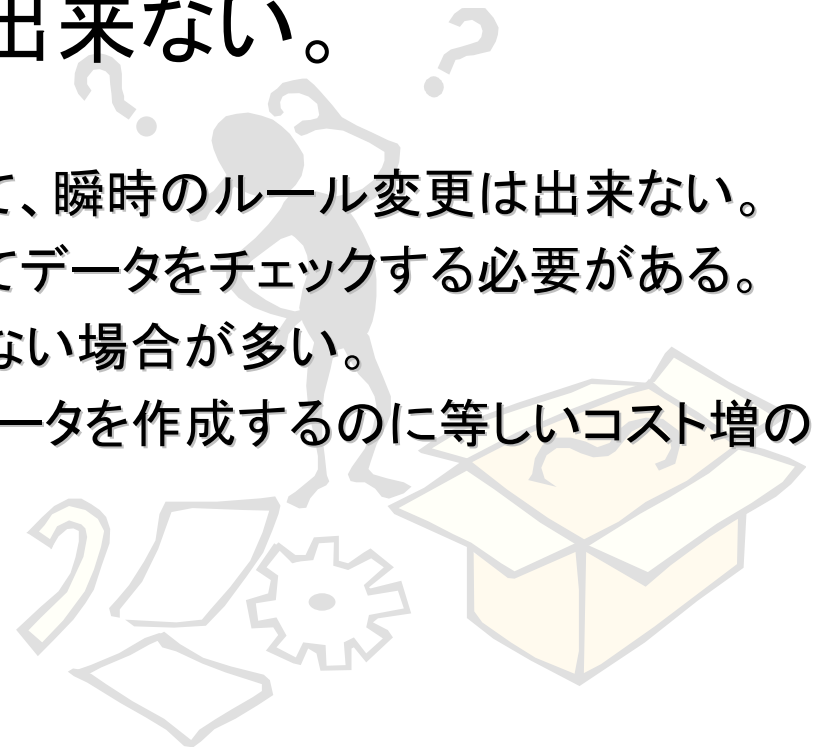
海外で作成する場合ローカライズも発生するが適切な言葉が見つからない。



問題点

- 安易なルール変更は出来ない。

徹底するまでに時間がかかりすぎて、瞬時のルール変更は出来ない。
ルール変更する場合、過去に遡ってデータをチェックする必要がある。
しかし、作った本人ですら覚えていない場合が多い。
従って「データの手直し」は1からデータを作成するのに等しいコスト増の
要因になりうる。

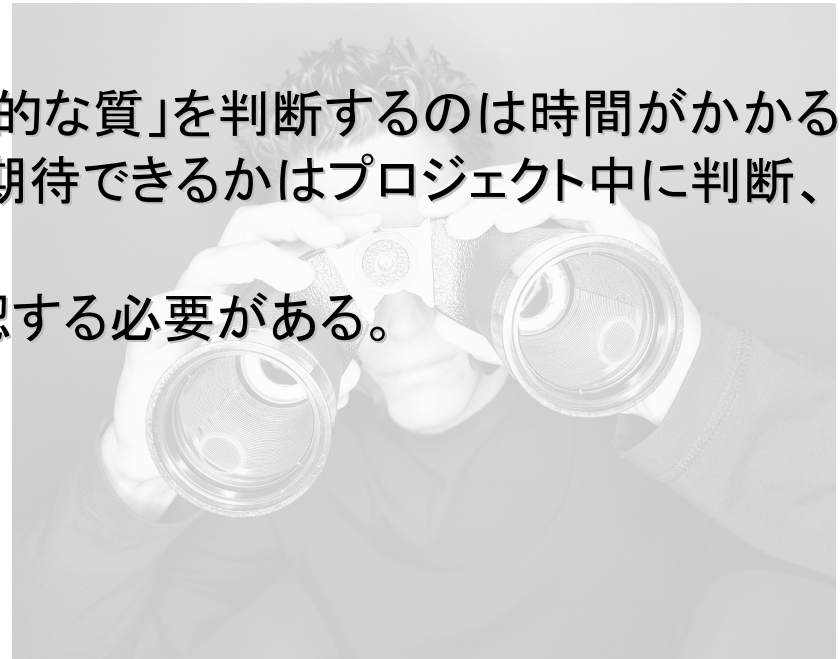


問題点

- 作業員のレベル予測が困難。

絵的な判断は容易、だが「データの質」を判断するのは時間がかかる
大量生産時に「データの質」を期待できるかはプロジェクト中に判断、
工数の修正を行う必要がある。

その為に、データの質を随時確認する必要がある。



問題点

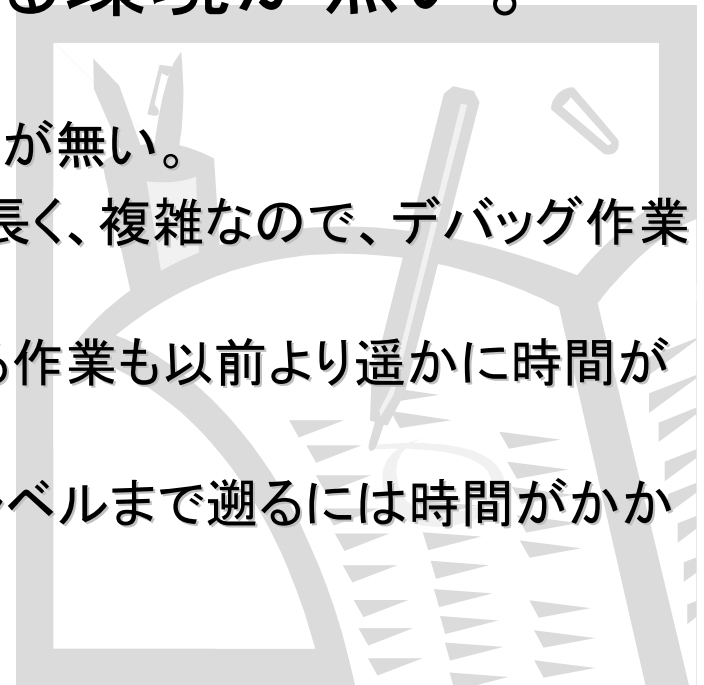
- 「データの質」をチェックする環境が無い。

処理速度的に最適なのか、判断する方法が無い。

実機での最終描画に至るパイプラインが長く、複雑なので、デバッグ作業が困難になった。

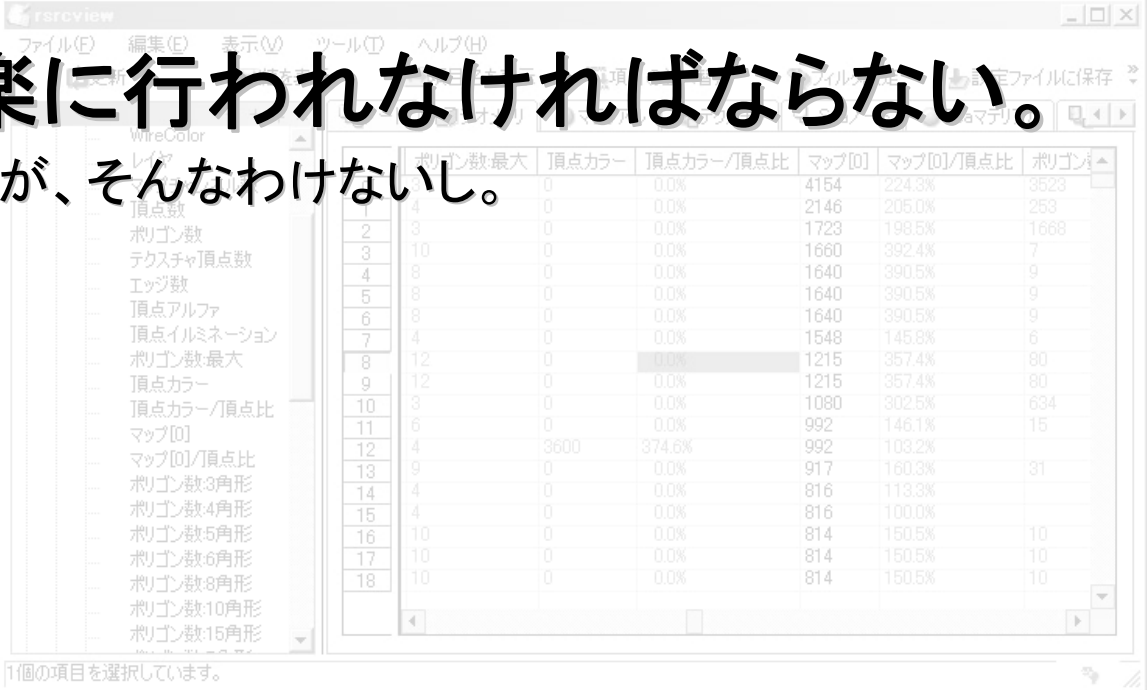
「コードミス」か、「データミス」か切り分ける作業も以前より遥かに時間がかかる様になった。

実機の解析ツールで分かっても、ソースレベルまで遡るには時間がかかりすぎる。



どうすればいい？

- **検収作業は楽に行われなければならない。**
人数が増えればいいが、そんなわけないし。



The screenshot shows a software window titled 'rsrcview' with a menu bar (File, Edit, View, Tools, Help) and a toolbar. On the left, there is a tree view of statistics. On the right, a table displays data for various statistics. The table has columns for 'ポリゴン数最大', '頂点カラー', '頂点カラー/頂点比', 'マップ[0]', 'マップ[0]/頂点比', and 'ポリゴン数'. The table is sorted by 'ポリゴン数最大' in descending order.

	ポリゴン数最大	頂点カラー	頂点カラー/頂点比	マップ[0]	マップ[0]/頂点比	ポリゴン数
1	4	0	0.0%	4154	224.3%	3523
2	3	0	0.0%	2146	205.0%	253
3	10	0	0.0%	1723	198.5%	1688
4	8	0	0.0%	1660	392.4%	7
5	8	0	0.0%	1640	390.5%	9
6	8	0	0.0%	1640	390.5%	9
7	4	0	0.0%	1640	390.5%	9
8	12	0	0.0%	1548	145.8%	6
9	12	0	0.0%	1215	357.4%	80
10	3	0	0.0%	1215	357.4%	80
11	6	0	0.0%	1080	302.5%	634
12	4	3600	374.6%	992	146.1%	15
13	9	0	0.0%	992	103.2%	15
14	4	0	0.0%	917	160.3%	31
15	4	0	0.0%	816	113.3%	15
16	4	0	0.0%	816	100.0%	15
17	10	0	0.0%	814	150.5%	10
18	10	0	0.0%	814	150.5%	10

どうすればいい？

- 一目でデータを理解しうる状態を作る必要がある。
UIを一杯並べると、一覧性がなくなる。



どうすればいい？

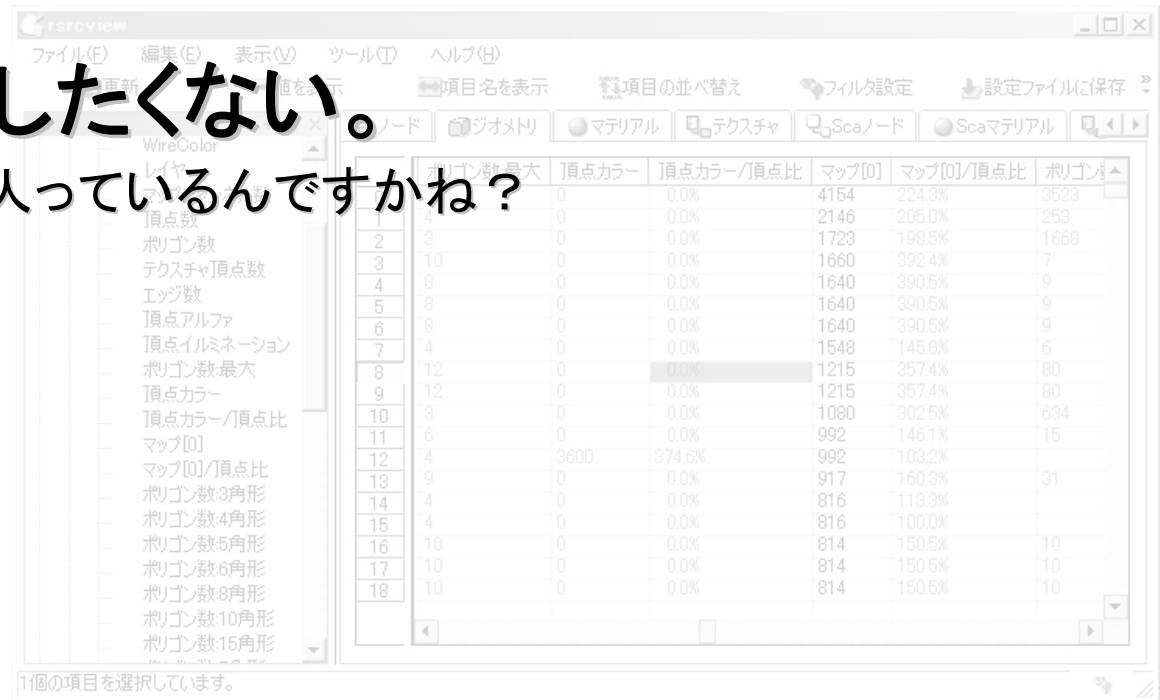
- **拡張は自由に行えないと不味い。**
要件はプロジェクト毎で多岐に渡る

The screenshot shows a software interface with a table of material properties and a list of material types on the left. The table has the following columns: Vセット数, マップ数, 頂点数, ポリゴン数, テクスチャ頂点数, エッジ数, 境界数, 法線数, 法線/頂点比, ポリゴン数4角形, UVセット名[1], and ポ. The table contains 21 rows of data. The left sidebar lists material types such as マテリアル, 名前, タイプ名, Color, Transparency, Ambient, Incandescence, BumpMapノード, BumpMapテクスチャ, Diffuse, Translucence, TranslucenceDepth, TranslucenceFocus, Colorノード, Colorテクスチャ, Transparencyノード, Transparencyテクスチャ, Ambientノード, and Ambientテクスチャ.

Vセット数	マップ数	頂点数	ポリゴン数	テクスチャ頂点数	エッジ数	境界数	法線数	法線/頂点比	ポリゴン数4角形	UVセット名[1]	ポ
0	2	609	366	863	677	未調査	977	1.919450	293	map1	
1	1	586	188	670	627	未調査	670	1.143345	88	map1	6
2	1	458	139	556	505	未調査	556	1.213974	133	map1	3
3	1	470	122	478	474	未調査	478	1.017021	112	map1	
4	2	215	125	463	333	未調査	275	1.279070	119	map1	2
5	1	184	149	457	331	未調査	301	1.635870	134	map1	
6	2	256	113	452	354	未調査	452	1.765625	113	map1	
7	1	256	112	448	352	未調査	256	1.000000	112	map1	
8	2	256	112	448	352	未調査	256	1.000000	112	map1	
9	1	178	128	422	294	未調査	314	1.764045	120	map1	
10	1	178	128	422	294	未調査	314	1.764045	120	map1	
11	2	172	110	418	264	未調査	416	2.418605	110	map1	
12	2	172	110	418	264	未調査	416	2.418605	110	map1	
13	2	172	110	418	264	未調査	416	2.418605	110	map1	
14	2	172	110	418	264	未調査	416	2.418605	110	map1	
15	2	172	110	418	264	未調査	416	2.418605	110	map1	
16	2	172	110	418	264	未調査	416	2.418605	110	map1	
17	1	123	96	360	216	未調査	360	2.926829	56	map1	2
18	1	90	77	308	161	未調査	200	2.222222	77	map1	
19	1	90	77	308	161	未調査	200	2.222222	77	map1	
20	1	90	77	308	161	未調査	200	2.222222	77	map1	

どうすればいい？

- ローカライズしたくない。
ローカライズ専門の人っているんですかね？



どうすればいい？

- **赤ペン先生機能が欲しい。**
間違ったら教えてよ。



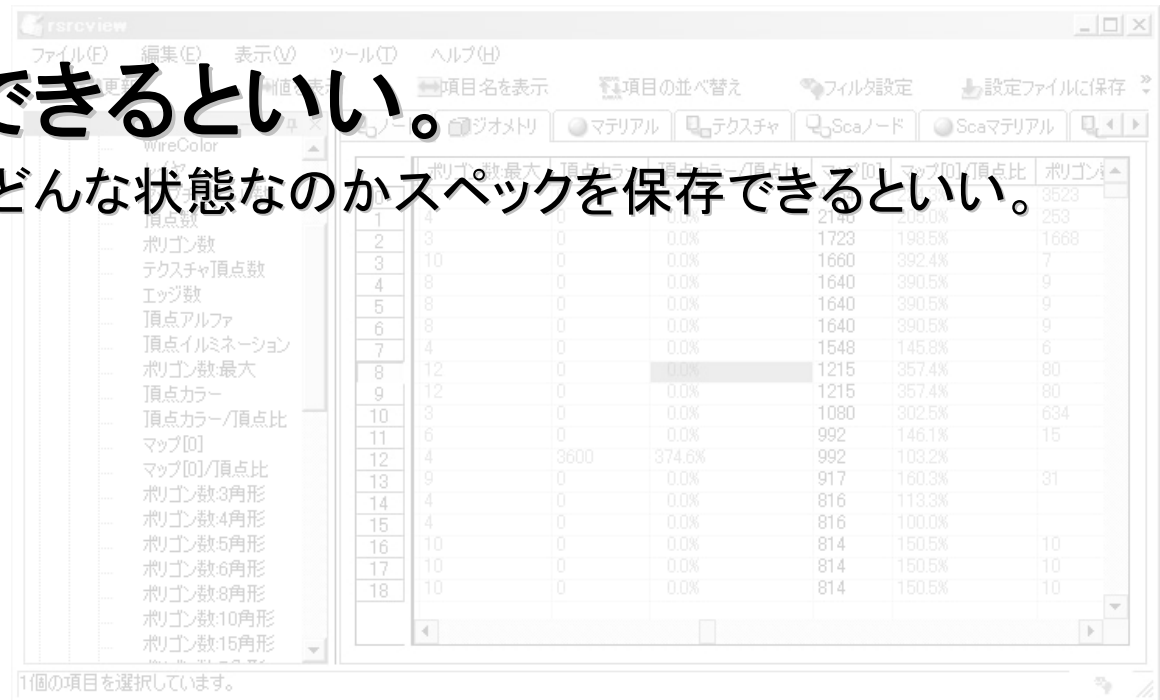
どうすればいい？

- **80%位は信用できるデータが出来ればありがたい。**
 そうすればチェックする範囲が少なくて済むから。

	Vセット数	マップ数	頂点数	ポリゴン数	テクスチャ頂点数	エッジ数	境界数	法線数	法線/頂点比	ポリゴン数4角形	UVセット名[1]	ポ
0	2	509	356	853	677	未調査	977	1.919450	233	map1		
1	1	586	188	670	627	未調査	670	1.143345	88	map1	6	
2	1	458	139	556	505	未調査	556	1.213974	133	map1	3	
3	1	470	122	478	474	未調査	478	1.017021	112	map1		
4	2	215	125	463	333	未調査	275	1.279070	119	map1	2	
5	1	184	149	457	331	未調査	301	1.635870	134	map1		
6	2	256	113	452	354	未調査	452	1.765625	113	map1		
7	1	256	112	448	352	未調査	256	1.000000	112	map1		
8	2	256	112	448	352	未調査	256	1.000000	112	map1		
9	1	178	128	422	294	未調査	314	1.764045	120	map1		
10	1	178	128	422	294	未調査	314	1.764045	120	map1		
11	2	172	110	418	264	未調査	416	2.418605	110	map1		
12	2	172	110	418	264	未調査	416	2.418605	110	map1		
13	2	172	110	418	264	未調査	416	2.418605	110	map1		
14	2	172	110	418	264	未調査	416	2.418605	110	map1		
15	2	172	110	418	264	未調査	416	2.418605	110	map1		
16	2	172	110	418	264	未調査	416	2.418605	110	map1		
17	1	123	96	360	216	未調査	360	2.926829	56	map1	2	
18	1	90	77	308	161	未調査	200	2.222222	77	map1		
19	1	90	77	308	161	未調査	200	2.222222	77	map1		
20	1	90	77	308	161	未調査	200	2.222222	77	map1		

どうすればいい？

- 履歴を保存できるといい。
過去に作った物が、どんな状態なのかスペックを保存できるといい。



データデバッグツール

どうすればいい?で
当日デモします。

落とし穴

- **意識が低いと役にたたない。**

ツールを作成しても意識が低いと役にたたない。

だから、なにはともあれ作成ルールを守る意識を高める努力をする。

努力とは数値で示す事だ。

具体的な事例を数値で示す事が重要。



落とし穴

例えば、↓の様になる。

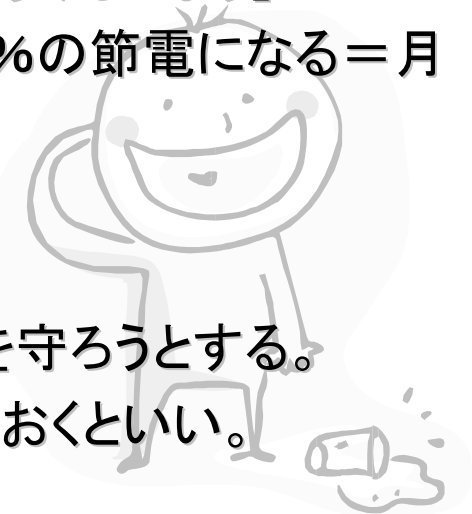
「消費電力を下げる為にエアコンの温度設定を1度あげましょう」

「冷房の場合、1度設定温度を上げただけで、約10%の節電になる＝月の電力代が6000円だとして600円安くなる」

どちらの表現が、それを実行しようとするだろうか？

具体的な数値でメリットを示された方が、人はそれを守ろうとする。

作成方法以外にも、やっていい事、悪い事を纏めておくといい。



本当はその前に

- 今までは「大量生産」に入った時の話。
- プリプロの段階で、データパイプラインをデバッグすれば、問題の解決は早くなる。
- しかしデータ出力後のデータデバッグは、デザイナーには困難である場合が多い。
- それ専用の3DCGツールみたいなのがあれば、デザイナーもデータデバッグできるのだが、、、。

本当はその前に

- 時間があればデモします。

