オープンソースを使った 3D データの活用の可能性

ソリューション本部　システム開発部
光山　萬珠

1. はじめに

近年、製造業においては、3DCAD による設計が標準になっており、データの受け渡しも 3D データにて行われている。ただし、この 3D データは CAD/CAM/CAE 以外で一般的に利用されるケースは少ない。これは、3D データが CAD に依存しており、そのままではデータのロードができなかったり、中間ファイルの仕様が難しかったりで、3D データを使ったシステムを開発することは難易度が高い。また、このシステムを開発する上でCADや開発ツールが必要となり、そのライセンス費も高額である事から開発者は限定されてしまう。これらの事がシステム化にかかる費用の高騰を招き、3D データが利用されにくい一因となっている。

3D データを活用できる一つの例として、パーツカタログがある。パーツカタログでは、部品イラスト(PDF 又は画像)と部品リストが存在する。特に部品イラストを作成するには、パーツカタログの管理者が 3DCAD のデータから複数のツールを介して PDF 又は画像にするが、この作業にとても手間がかかる。CAD データをそのままカタログに使えるのであれば、管理者の手間を大幅に削減できる上に、閲覧するユーザにとっても、3D で確認することによって製品をイメージし易いというメリットが生じる。

本稿では、当社の Web パーツカタログの製品である PLEX のパーツリストの表示において、CAD データを使った 3D データの表示がどこまで安価に実現できるかを調査する事とした。

2. 3D 技術とデータの選定

パーツリストを表示するまでには図 1 の様に、管理者が「①元となる図面や部品のデータを格納」し、「②格納したデータの紐付けなどの調整」を行い、一般ユーザが「③閲覧する」という流れとなる。

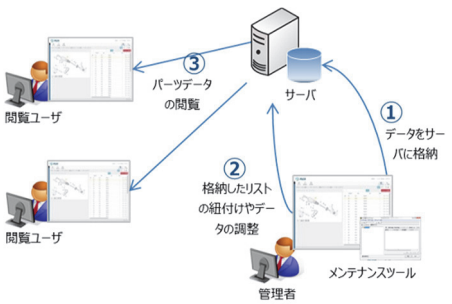

図 1　パーツリスト作成～閲覧の手順

それぞれに対する機能として、①では「CAD データから必要なデータを抽出する機能」、②では「抽出したデータと部品リストとの関連付けや位置調整するための機能」、③では「Web ブラウザ上に表示して確認する機能」が必要である。この条件に合致し、オープンソース又はツールやライブラリが安価である事を条件として、調査の方を進めた。

2. 1　3D データフォーマットの選定

3D データを表示するためには、CAD データから表示データ(ポリゴンデータ)を取得する必要がある。一般的な 3DCAD のファイルフォーマットは公開されていないため、このままでは取得するのが難しい。

そのため、3DCAD から出力可能であり、規格がオープンなファイルフォーマットの選定を行った。ファイルフォーマットしては、STL /VRML/IGES/STEP/JT などが存在する。STL と VRML は表示データを格納するためのフォーマットである。IGES/STEP/JT は他 CAD との形状データをやり取りするための中間ファイルフォーマットである。

それぞれのファイルフォーマットの中で、サポートされている 3DCAD の多さ、アセンブリ構造が定義されているか否か、表示データが定義されているか否かを評価対象とし、全てが（一部条件付きではあるが）可能な JT ファイルを選定した。

表 1 対応 CAD と出力データ比較表

	対応 CAD	ｱｾﾝﾌﾞﾘ	表示ﾃﾞｰﾀ
STL	○	×	○
VRML	△(※①)	×	○
IGES	○	×	×
STEP	○	○	×
JT	△(※①)	○	○

① 標準で出力可能な 3DCAD は多くないが、オプション製品や他社製の変換ツールが存在しているため、△としている。

2. 2 3D 表示技術の選定

プラットフォームに依存せずに 3D 表示を行う技術は、主に”3D PDF”、”WebGL”、”JavaApplet による OpenGL”、”Flash”の 4 種類の技術がある。今回はプラグイン等のインストールが不要、多くの Web ブラウザ上で 3D を表示可能、有償なツールは不要、開発における情報量も多い WebGL という技術を選定した。

表 2 3D 表示技術の比較表

	Web 化	無償ﾂｰﾙ	情報の多さ
3D PDF	×	×	×
WebGL	○(※①)	○	○
OpenGL	△(※②)	○	○
Flash	△(※②)	×	△

① ブラウザに依存するが、最新のバージョンであれば大抵の種類のブラウザで動作するため○としている。

② ブラウザの他にプラグインのインストールが必要であり、モバイル端末ではブラウザのバージョンによっては表示できないため△としている。

3. データの調査方針の決定

前章で 3D データのフォーマットと 3D の表示技術の選定を行った。次に具体的な「JT ファイルからの必要な情報の抽出」と「WebGL の可能性」の調査方針を下記の様に決定した。

3. 1 JT ファイルからの必要な情報の抽出の方針

JT フォーマットとは、2012 年に ISO に承認された 3D の軽量なファイルフォーマットである。形状データの他に、表示データ、色情報、PMI 情報、その他各種属性など、多くの情報を持つことができる。

3D の表示を行うためには、最低でも表示データである頂点情報を取得する必要がある。情報の取得を容易にするため、JT ファイルを読込むオープンソースとして、2 つのライブラリを見つける事が出来たが、ISO に承認されているバージョンである Ver9.5 に対応した jcadlib という Java のライブラリを使用する事とした。

また、調査としては ISO 14306 の規格を購入し、必要に応じてライブラリのコードを変更して調査を行った。

3．2 WebGL の可能性の調査方針

WebGL は Web ブラウザ上で 3D グラフィックを高速に表示するための標準仕様で、Javascript で記述する技術である。純粋な WebGL は多くの手続きが必要であり、行列やベクトルなどを十分に理解している必要がある。そのため、今回はこの手続きや面倒な計算を省くため、WebGL のオープンソースのライブラリの中で有名な three.js を使用する事にした。また、パーツリストに必要と思われる機能をリストアップし、一つずつ実装しその具体的な手法も含めて調査することにした。調査対象とした選定した機能のリストは以下の通りである。

閲覧ユーザ向けの表示機能として、

① 3D データ表示とマウス操作によるビューイング
② アイテムの選択とハイライト
③ 部品記号用記号用のバルーンの表示
④ 部品の分解アニメーション

管理者向けの機能として、
⑤ 部品位置の調整
⑥ アセンブリツリーの表示と操作

4．調査結果

4．1 3D データからの必要な情報の抽出の結果

JT ファイルの構造は図 2 に示すとおりである。ファイル構造としては大きく分けて、1 つのファイルヘッダ、1 つの TOC セグメント、複数のデータセグメントがある。ファイルヘッダには、JT ファイルのバージョンや TOC セグメントの開始位置が設定されている。TOC セグメントには、データセグメントの開始位置、定義サイズ、タイプが設定されている。データセグメントには、アセンブリ情報、形状情報、表示情報、プロパティ情報など各セグメントの種類に応じて設定されている。

データセグメントの中で最も重要なのが LSG セグメントである。LSG セグメントの中に個々の情報が Node としてわかれており、子 Node を紐づけていく事でアセンブリのツリー構造が構築できる。

Node には複数のタイプがあり、インスタンスを表す node(Instance)、実体を表す node(MetaData や Part)、表示データを表す node(TriStripSetShape)などがある。これらの Node と Node が持つ情報から、部品の頂点データ、法線データ、Index 配列などの表示データや部品に付加された情報を取得する事が出来る。また、各インスタンス Node に座標変換情報や、マテリアル情報を持つため、位置情報や色情報も取得する事が出来る。

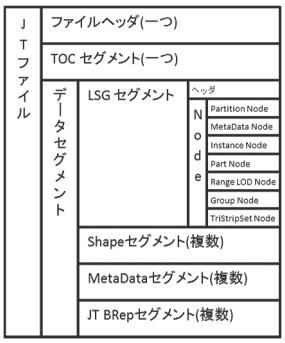

図 2 JT ファイル構造イメージ

簡単にデータ構造について説明したが、jcadlib を使用すればソースコードを大きく変更しなくても、これらの情報は取得可能である。必要な情報だけ出力するように変更する事で、今回必要最低限の情報を抜き出す事が出来た。ただし、属性情報については、一部処理が実装されていないため、独自に実装する必要があった。

4．2 WebGL の可能性の調査結果

① 3D データの表示とビューイングにおいては、4.1 で抽出した表示データを three.js の表示

データに変換するところだけが厄介であるが、それ以外はサンプル通りに実装は可能である。JT ファイルから抽出した頂点配列と法線配列には、それぞれ別々の index 配列が存在する。同じ頂点や同じ法線は 1 度しか定義しない事でファイルサイズを小さくするためである。ただし、three.js では頂点と法線は 1 対 1 対応であり一つの index 配列しかないため、配列を組みなおさないといけない。その後、変換した頂点配列、法線配列、index 配列を three.js の形状情報を示すオブジェクトの BufferGeometry にセットする。また、three.js の色や質感情報を示すオブジェクトの MeshPhongMaterial に、色・反射光・輝度・透明度のパラメータをセットする。この二つのオブジェクト、three.js 用の描画オブジェクトである Mesh を作成する事が出来る。最後に、Mesh が保持している位置情報行列に対して、抽出した位置情報行列をそのままセットする事で 3D の表示データの作成は完成である。これを全ての部品に対して行う事で、概ね 3DCAD と一致したデータを表示する事が出来た。また、マウス操作によるビューイングについては three.js のサンプルに付属している TrackballControll.js を使い簡単に実装ができる。マウス操作を行うコントロールは他にも存在するため、必要に応じて使い分けると良い。

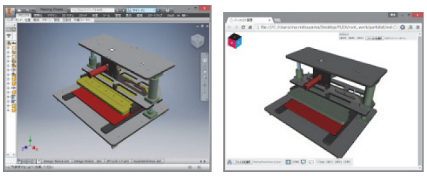

図 3 左は 3DCAD、右は three.js

② アイテムの選択については、Raycaster オブジェクトを使用し簡単にクリックした位置にあるアイテムの取得を行う事ができる。これは three.js の公式のドキュメントに書かれているサンプルの通りである。その後、取得されたアイテムである Mesh のマテリアルの色を変更する事でハイライトを行った。

③ バルーンは、部品の ID を示す”記号”と、”記号と部品を繋ぐ線”が必要である。”記号”については HTML に文字入りの DIV 要素を追加し、線については 3 次元の要素として作成する事とした。”記号”については HTML 上の部品から少し離れた位置に、DIV 要素を Javascript の DOM 操作にて配置し style にて位置設定を行った。”記号と部品を繋ぐ線”の作成は少し難しい。この線自体は 3D である。始点は部品の原点とすることで問題がないが、終点は”記号”の位置に合わせる必要がある。”記号”の位置は HTML の座標(2D)であるため、その変換処理を行う必要がある。通常であれば、複雑な処理になるが、Vector3 の unproject メソッドで簡単に変換が可能である。

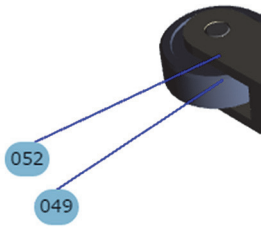

図 4 バルーンのイメージ

④ 部品の分解アニメーションは、自動で分解方向を認識するのは難しい。そのため、今回は部品ごとのアニメーション時間と分解方向を外部ファイルで定義できる仕組みを作成し、実装を行った。定義する手間は発生するが、同じパターンの分解においては同じファイルを使用して分解する事が可能である。アニメーションは一定時間に一定量部品を移動させる処理で実

装した。この部品の移動処理は、Mesh の applyMatrix というメソッドで平行移動行列を掛け合わせる事で実装可能である。

図 5 部品のアニメーション後

⑤ 部品位置の調整については、マウスの移動量を 3D の移動量に変換するだけであるため、バルーンの処理で行った 2D 座標から 3D 座標の変換(unproject)を開始点と終了点に対して行い、その 2 点から出来るベクトル方向に移動する事で実装は可能である。移動処理は分解アニメーションと同様の処理である。

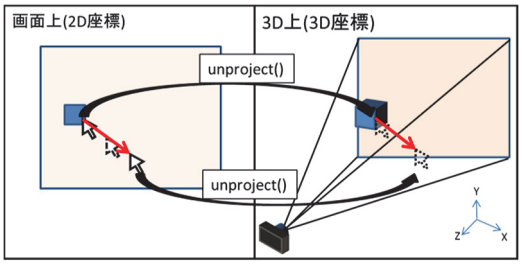

図 6 マウス移動量を 3D の移動量

⑥ アセンブリツリーは、jquery.treeview.js を使用して簡単にツリー構造の構築ができる。JT フォーマットでは複数の Node があることは説明したがこの中でツリーに表示するのは、ルートを表す PartitionNode と各アセンブリや部品のインスタンスを表す InstanceNode を表示し、アセンブリならフォルダ、部品ならファイルとして表示する。その後、ツリー上の項目と 3D の部品の紐付けは Javascript で行い、お互いにマウスでクリックされればハイライトする仕組みを追加した。また、このツリー上にて部品の表示非表示やプロパティの表示などの管理者として必要となる機能も実装した。

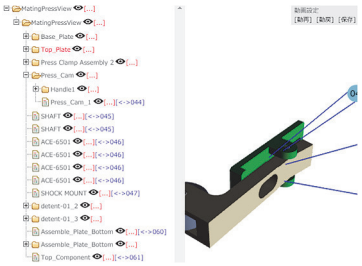

図 7 アセンブリツリー

5. 課題

今回の調査で大体は想定通りではあったが、大きな物としては 2 つの課題があった。

① jcadlib にて JT ファイルの情報を取得しているが、実際にはファイルの中の一部の情報しか抽出できていない。例えば、線分、点アイテム、PMI 情報や形状情報、光源やテクスチャ情報は取得できていない。また、Ver.9.5 以外のバージョンには対応できていない。その上、JT ファイルは CAD ファイルと同様にアセンブリファイルと部品ファイルとして複数の JT ファイルに分ける事ができるが、そのケースにも対応できていない。これらについては、jcadlib に対して機能として追加していく事で対応可能ではあるが、データ構造を十分に理解しておく必要があり、多くの工数がかかってしまう。本格的にこれらの対策を行う場合は、有償のライブラリを使う事で、この作業を軽減することが可能である。プロジェクトの規模に応じて、有償版、無償版のどちらを使用するを決定するとよい。

② 本稿では JT ファイルを出力する 3DCAD として Autodesk® Inventor®を使用したが、CAD で設定した情報の中で出力されていない物があった。例えば CAD 上で変更した部品の名前や、部品のプロパティに関してはほとんど出力されなかった。抽出可能な属性としてはカスタム設定した属性のみであった。このため、DB に登録されているパーツリストの品番と CAD に設定された部品番号の自動紐付けは、JTファイルだけでは難しい。そのため、対策としては使用する CAD ソフト毎に必要に応じてプラグインを作成し、必要なデータは別のデータとして出力する様な仕組みを追加する必要があるかもしれない。この仕様は使用する CAD ソフトに依存する可能性が高いので、事前に対象となる CAD 毎に調査した方が良いかと思われる。

6. まとめ

本稿では開発ツールやライブラリは全て無料の物を使用した。課題以外に関しては大きな問題もなく概ね当初の目的を達成する事ができた。今回使ったオープンソースのライブラリとしては、JT ファイルからの抽出用に jcadlib、3D 表示用に three.js、HTML の DOM 操作のために jquery、アセンブリツリーの表示用に jquery.treeview を使用している。これらは全て MIT ライセンスであり、とても使いやすいライブラリとなっている。開発ツールについてはテキストエディタとブラウザの開発者ツールしか使用していない。

今回はパーツリストに対する調査を行ったが、CAD データから抽出した表示データを Web ブラウザ上に実装する事で、3D データの活用可能なシーンは多い。例えば、PLM との連携、エンドユーザ向けのリアルな製品イメージの公開、自社のグループウェアへの組み込み設計状況の確認などが考えられる。

多くの枠組みで使用可能にするためのモジュール化や情報共有を行っていき、今後 3D データの活用拡大に向けて取り組んでいきたい。

＜参考文献＞

1) 「three.js による HTML5 3D グラフィックス」(2013 年 10 月 10 日, 株式会社カットシステム)
2) 「ISO 14306:Industrial automation systems and integration – JT file format specification for 3D visualization」(2012 年 12 月 15 日)
3) 「three.js」
http://threejs.org/
4) 「jquery」
https://jquery.com/
5) 「jquery-plugin-treeview」
http://bassistance.de/jquery-plugins/jquery-plugin-treeview/
6) 「jcadlib」
https://github.com/Liamsi/jcadlib