２．三次元アーカイブスの構築と利活用の手引き

はじめに

本章では、様々な資料を整理・加工して過去に構築された三次元アーカイブスが、どのような形で利活用可能かをまず示す(２－１、２－２)。

次に、将来の様々な利活用を可能とする記録保存データの整理・加工方法について、この研究の中で実行した手順を解説する(２－３)。

更に、本処理系の最大の特徴である、任意形式の保存データに添付して、解読処理を処理系に指示するための形式定義ファイル（メタファイル）の作成方法について解説し（２－４)、これを記録データに添付して保存する方法について解説する(２－５)。

最後に、データベース上でデータを保管すると共に、任意形式のデータとして取り出す「三次元データ保管庫」の操作・運用方法について解説する（２－６、２－７)。

２－１．タブレットを用いた現場での閲覧機能(VC-3M)の利用

本節では、利活用形態の一つである、スマートフォンまたはタブレットを用いて現場での記録の閲覧を行うエンドユーザーのための操作方法の解説を行う。

このプログラムは、端末の背面カメラが取得した画像を表示画面にそのまま表示すると共に、それに重ねて記録保存された建造物の三次元データを CG で合成表示する。その際に、端末に装備された GPS センサ、加速度センサ、磁気センサから計算した端末の位置とカメラアングルを用いて、記録データの透視図を作成することにより、その建物が存在していた位置あるいは将来計画されている位置に存在するかのように合成表示することが特徴である。写真 2-1,2 は、平成 26(2014)年 10 月８日に、奥尻島の岬公園において、平成 5(1993)年以前に存在していた集落を表示したタブレットを見ながら、小学生たちが過去の集落の道を歩いて追体験している風景である。「五区」と呼ばれていたこの集落は、平成 5(1993)年７月 12 日に発生した北海道南西沖地震とそれに伴う津波で消滅し、生き残った住民は「防災集団移転」により高台に新たに開発された団地に全て移転した。その跡地は公園として整備され、現在ここには住宅はない。昭和６(1931)年に建立された「徳洋記念碑」が変わらず存在してランドマークとなっている。

写真 2-1 奥尻島における体験教室の風景（2014.10.8）

児童が閲覧している町並のアーカイブスは、被災前の古写真から復原した個々の住宅を

道に沿って配置して再現したものである。

タブレット端末の操作は、基本的には眼前にかざしながら持って歩くだけであり、体験教室で試した結果、小学校低学年でも十分に、起動から終了までの一連の操作を理解し実行できることを確認した。子供向けに、「むかしめがね」という名称で説明した。

写真 2-2 タブレット端末の表示内容

（1）起動と表示する場所の選択

図 2-1-1 マイアプリ画面で、「むかしめがね」のアイコンをタップして起動する

使用したタブレットは、OS として Android を搭載しており、起動した状態で、主画面にアプリのアイコンが表示されている。画面上で、アイコンを指で軽く叩いて（以後、「タップ」して)、「むかしめがね」アプリを起動すると、次のような初期画面が表示される。

図 2-1-2　「むかしめがね」初期画面

「見たい場所」のボタンをタップすると、準備されているいくつかの場所から選択を行う画面に進む。「おわり」を押すと、終了して、OS の画面に戻る。

図 2-1-3　見たい場所の選択画面

見たい場所の一覧から、場所を選び、文字をタップすると、合成表示の画面に進む。

（２）GPS 衛星の電波取得待ち

合成表示画面では、背面カメラから取得した画像が表示され、その上に記録保存された建物などが合成表示される。

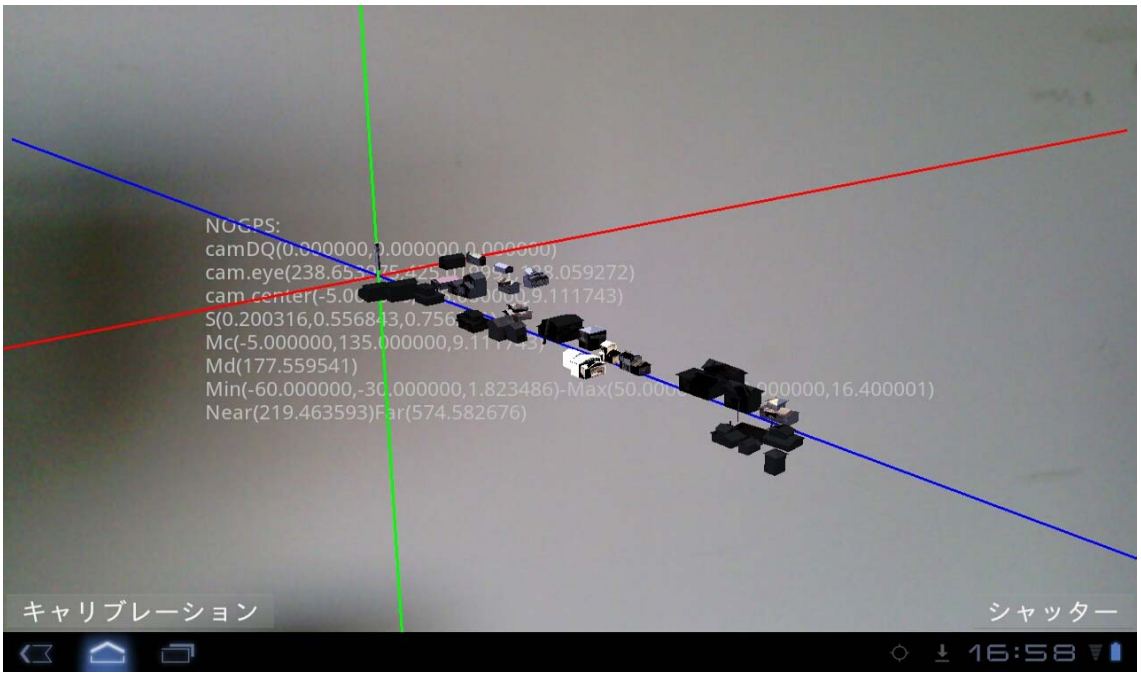


図 2-1-4 GPS 衛星を探索中の表示

ここで GPS 衛星からの電波から現在位置が取得できるようになるまでの数秒～数十秒の間は、端末の位置にかかわりなく、建物を表示する。その時、端末の向きだけは表示に反映する。例えば、端末を地面に向けると、建物の屋根を上から見下ろした表示となる。また端末を垂直に北に向けると、建物の南面を表示する。横に回転させると、建物が回転し、すべての壁面を見ることができる。

GPS 衛星からの電波が受信できない場所で動作させた場合には、この状態が続く。

（３）現場での閲覧

GPS 衛星から位置が取得できると、その位置からの表示に変わる。この時、町並の中に立っていれば、道の両側に建物の正面が並んで表示される。道に沿って進むと、手前の建物が後ろに隠れ、遠くの建物が近づいてくる。

建物の内部に入ってしまうと、建物の内部の壁だけが表示され、背景の風景は全て隠されてしまう。

町並から離れた位置から眺めた場合には、端末の向きが悪いと建物が全て範囲外となり、画面には背景だけが表示されることがある。そのような場合には、画面の中央に Left（ひだり）、 Back（うしろ）, Right（みぎ）といった文字が表示される。

Left（ひだり）が表示された場合には、端末を左に向けると、町並が表示の範囲に入ってくる。Back（うしろ）の場合には、回れ右をすると町並が見つかる。

Far（とおい）と表示されるのは、遠すぎて町並全体が画面中央の小さな点にしかならな

いような場合である。例えば東京からむかしめがねで、奥尻の町並を見ようとした場合には、北を向けても Far(とおい)と表示されるだけである。

（４）足による補正（高学年向き）

GPS センサの精度が低い場合には、町並が表示される位置がずれる。そのような場合には、左下の「キャリブレーションボタン」を押すと、町並の表示位置が画面に固定される。そこで、このまま端末を持って歩いて移動し、正しい位置を見つけ、そこでボタンを離す。すると、その間のずれの距離が記憶され、以後の表示はこれを用いて（補正して）表示される。GPS センサが衛星電波を受信し続けている間は、比較的滑らかに位置情報が得られる。しかし、一度 GPS センサを停止し、再起動すると、前回とは大きく異なる位置情報が得られる場合がある。従って、閲覧を開始してから補正を行うと、閲覧継続中は比較的正確な位置に表示される。

図 2-1-5　キャリブレーションボタン

この補正は、携帯端末の移動と回転の両方に関して同時に行うため、押し始めと離すタイミングで端末が同じ向きである必要はない。また、一度補正を行った状態で、さらにズレがある場合には、再度補正を行うと、補正は累積的に行うため、補正の操作を繰り返し正しい位置に近づけていくことができる。

（５）シャッターによる静止画の記録

うまく表示されている状態で、好きな場所が見つかった場合、そこで右下の「シャッター」のボタンを押すと、表示されている画像をデジタルカメラと同じように JPEG ファイルに保存すると共に、その時の端末の位置と姿勢を撮影記録として保存する。

図 2-1-6 シャッターボタン

画像の保存が終了すると、「保存しました」というメッセージが一瞬表示され自動的に消える。

（６）屋内での記録データの再生

現場での表示は、下の「戻る」ボタンにより終了し、最初の画面(1)に戻る。そこで、「おわり」のボタンをタップすると、アプリが終了する。この時に、シャッターで記録したデータがファイルに保存される。「おわり」を押さないまま、バッテリー切れ等により終了したような場合には、記録は残されない。

図 2-1-7　「歩いた場所」をタップして記録データを再生する

現場表示画面から戻った初期画面または、再びアプリを起動した時の初期画面で、「歩いた場所」のボタンをタップすると、シャッターで記録した場所の一覧が、小さな画像で表示される。

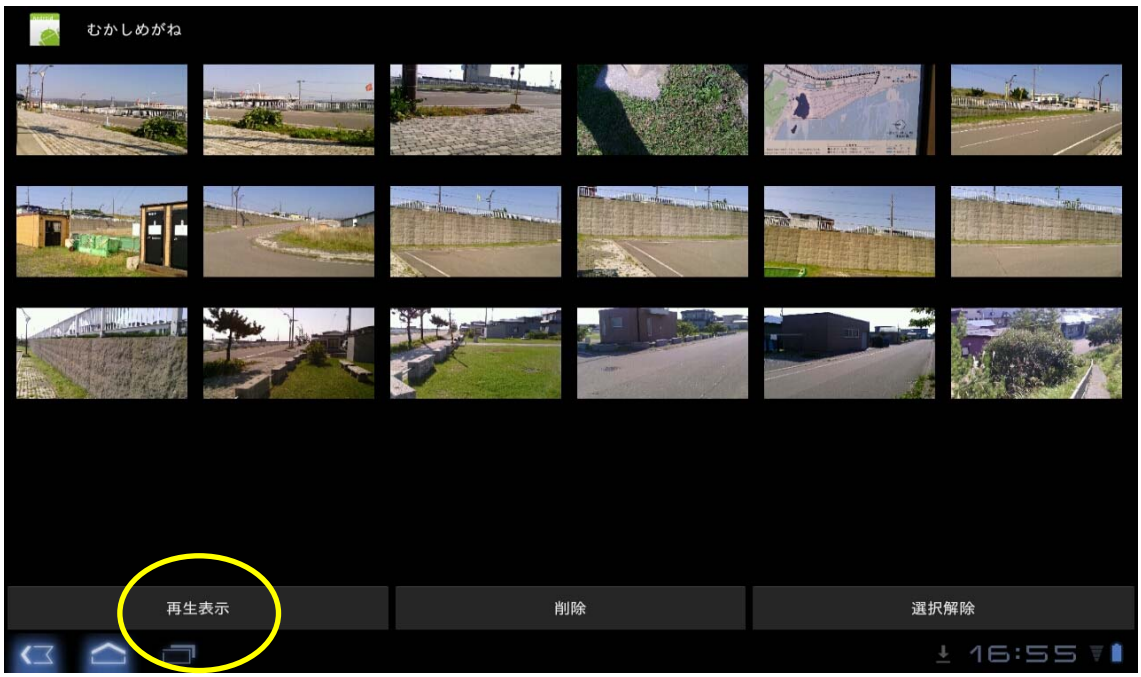


図 2-1-8　記録した場所の一覧（サムネイル画像）

この画面に並んで表示される小さなサムネイル（見出し）画像は、保存した背景画像だけが表示され、建物などのデータの合成表示はない。シャッターを押した時点で、建物等

が表示範囲の外であった場合には、サムネイル画像の両側に赤い線が表示される。

この一覧から、ある画像を選択すると、その画像の両側に緑色の表示が出て、選択された状態であることを示す。

この画像が一つ選択された状態で、「再生表示」ボタンをタップすると再生表示撮影した時点での合成画像が表示される。この再生表示画面は、視点位置が保存された画像に固定されているため、携帯端末を動かしても表示は変化しない。

一つのサムネイル画像を選択した状態で、「削除」ボタンを押すと、その画像と、カメラ位置・姿勢等の記録が削除される。（画像ファイルそのものは残している）

（７）その他

タブレットを使用する上で有用ないくつかの操作方法（ノウハウ）について解説する。

・画面キャプチャ（静止画）

広報用資料や解説資料を作成するために、画面のキャプチャないしスクリーンショットを作成したい場合がある。PC 上の Windows においては、PrtScr キーを押すことにより、クリップボードと呼ばれる一種のバッファに画面データがコピーされ、次に Ctrl－V のキー操作により、別のアプリケーション等（例えばワードプロセッサ）の上にこの画像を貼りこむことができる。類似の操作を Android タブレットで行う場合には、機種によって異なるが、例えば画面左下の「戻る」と「ホーム」のソフトボタン（画像で表示されているボタン）を同時に素早くタップすると、保存するファイル名を入力するダイアログが開き、画像ファイルとして画面を保存することができる。あるいは、「電源」と「音量(-)」のハードウェア・キーを同時に長押しすることにより、同様の操作ができる。作成した画像ファイルをワープロ原稿等に使用することができる。OS のバージョンにより操作方法が異なるため、機種毎に確認する必要がある。

・画面キャプチャ（動画）

HDMI 端子を有する携帯端末の場合には、接続ケーブルでモニタ等に接続することにより、表示されている映像を取り出して複製することができる。但し、コピー・プロテクトがかかっている機種の場合には、これを動画としてファイルに保存することはできない。

専用アプリをセットアップして、動画を保存する方法がある。操作マニュアルを動画として作成するような目的に使用可能である。

２－２．指導員のための手引き

指導員は、必要な携帯端末にプログラムとデータをセットアップし、起動できるような状態を準備する。次に、使用する携帯端末が、必要とする機能と十分な精度を有しているかをテストする。２－１で解説したエンドユーザーによる閲覧が終了した後、必要であれば撮影データ等の回収・保存を行う。最後にアンインストールを行い、携帯端末をセットアップ前の状態に戻す。

エンドユーザーが本節に解説する作業を自ら実行してもよい。

使用する携帯端末は、操作時点でインターネットに常時接続している必要はない。

（１）セットアップの準備

「むかしめがね」を携帯端末にセットアップするためには、必要なファイルを携帯端末の内蔵 SD カードにコピーした上で、VC-3M.apk というセットアップ・プログラムを実行する必要がある。

セットアップ・プログラムを実行するためには、内蔵 SD カードに存在するファイルを表示し実行するための、「ファイル・マネージャ」ないしこれに類するプログラム（Windows の「エクスプローラ」に相当するもの）が必要である。機種によってはプレ・インストールされているが、そうでなければ、フリーソフトをダウンロードしてもよい。

（２）必要なファイルのコピー

セットアップを行うためには、 WiFi(無線 LAN によるインターネット接続)経由でインターネットに接続してダウンロードする方法、USB 経由でパソコンに接続して必要なファイルをコピーする方法、あるいは USB メモリを接続して、そこからファイルをコピーする方法などがある。

コピーするファイルの内、大きな容量を占めているのが町並みを構成する建物等のデータである。これに、必要なプログラムと、「見たい場所」の一覧を表示するためのリストなどがある。これらは、「VirtualConverter」という、SD カードのルート直下のフォルダに格納されている。

WEB 経由でダウンロードしたセットアップ一式は、WiFi 経由でインターネット接続した携帯端末でダウンロードするか、またはこれをダウンロードした PC から USB 経由で携帯端末にコピーする。

セットアップ一式は、ZIP 形式で一つのファイルに束ねられているため、これを解凍する必要がある。この解凍は携帯端末上で行っても、また PC 上で予め解凍してからコピーしても良い。

（３）セットアップ

「むかしめがね」をセットアップするためには、「設定」で、商用ソフト以外のプログラムのセットアップができるようにしておく。セットアップを実行するためには、上記の「ファイル・エクスプローラ」で、VirtualConverter ディレクトリを開き、その中にある、VC-3M.apk というファイルをタップする。古いバージョンの VC-3M が既にセットアップされている場合には、上書きでセットアップすることができる。

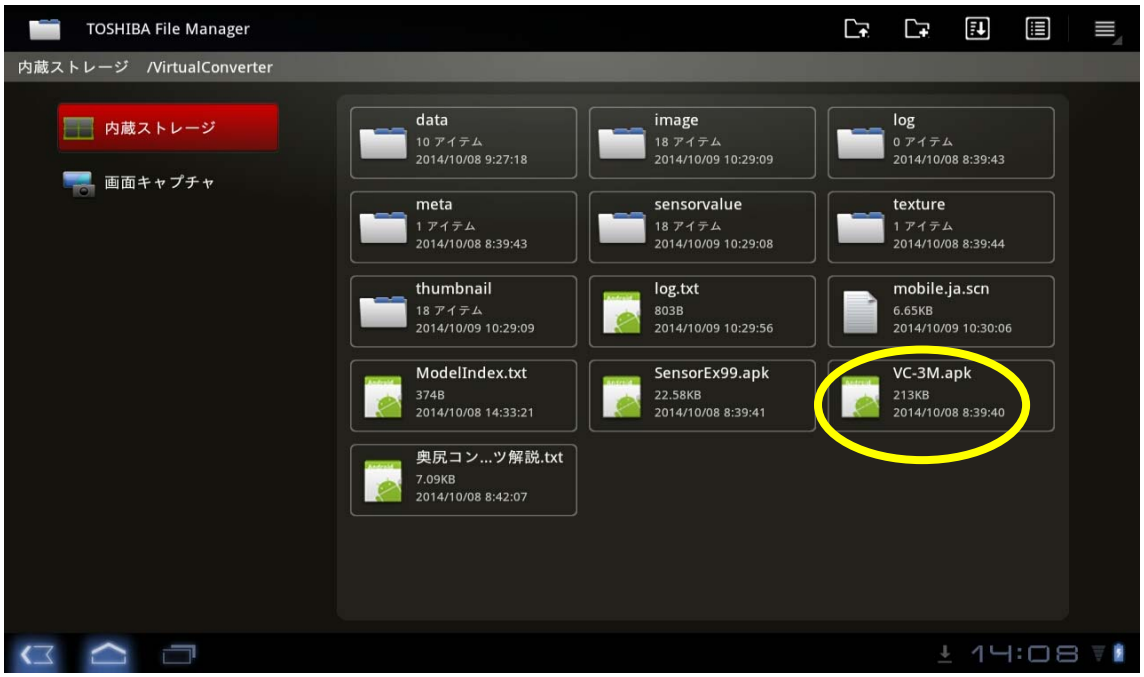


図 2-2-1 VirtualConverter ディレクトリ内部のファイル構成

（４）GPS 機能の設定

「むかしめがね」は、GPS を使用するので、「設定」画面で、GPS を使用できるように設定しておく。

この設定が行われていない場合には、アプリを起動した時点で、警告表示を行い、GPS が利用できない状態のままでアプリを実行する。その場合、GPS 電波を検索中、あるいは GPS を受信できないコンクリート造建物の中や鉄板屋根の下で使用した場合と同様の、端末の向きだけを用いて町並の全体を様々な角度から眺めるような表示動作を行う。

なお、GPS 機能は電力を消費するため、現地での閲覧等が終了した後には、再び「設定」画面を開いて、GPS のスイッチを切っておく方が、バッテリーが長持ちする。

（５）端末の精度の確認

GPS 衛星からの電波が利用できない状態において、加速度センサと、磁気センサの精度をテストすることができる。

加速度センサ（重力センサ）は、端末が静止した状態では、下を指す。しかし、手で持って歩き回る状態では、手振れにより加速度の向きは揺れ動いている。「むかしめがね」アプリでは、手振れの影響を緩和するために１秒程度以前までの過去の計測値の平均を求めて使用している。

磁気センサ（電子コンパス）の計測値は、携帯端末自身の帯磁によって影響を受ける。この影響は、携帯端末に固定した座標系では XYZ３軸の計測値に固定値を加えた値となる。従って、携帯端末を３軸に回転させることにより、その値を計測し、補正することができる。このような補正機能を備えた携帯端末の場合には、ユーザーの利用に先立って、端末自体を大きく振り回して回転させることにより、改善することができる。木製テーブルの

上などで、水平に 1 回転、横に 1 回転、縦に 1 回転する。

地磁気をベクトルとして見た時、水平ではなく地面と一定の角度を有しており、その水平成分が磁北を指している。このため、「むかしめがね」VC-3M では、加速度センサで検出した「下」の向きと直交する成分を抽出して、水平な北の向きとしている。

磁気センサが検出する磁場は、地磁気の他に建物や工作物の帯磁した鉄製品等の影響も受ける。また直流で送電している電車（地下鉄も含む）の架線などから生じる磁場の影響も受ける。例えば、無料でダウンロードできる「GPS Status」のようなアプリを起動し、現場付近で同じ場所で磁北の変化を観察したり建物内外で持ち歩いて磁北と建物の関係を見たりすることにより、影響を把握することができる。

携帯端末のセンサが計測する、携帯端末の座標系(UVW)に即した磁場の向き M と、重力の向き G を用いて、重力の向きに直角な磁北の向き（Y）と上（Z）を求める。

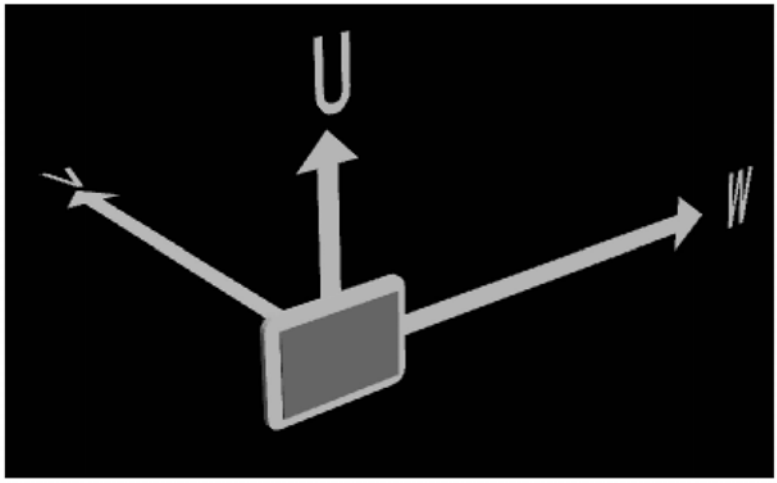


図 2-2-2 携帯端末の向きの取得

GPS 衛星からの電波からの位置（緯度、経度、標高）の計測ができるようになるまでの所要時間は、機種によって大きく異なる。一般に、過去の計測結果を記憶しておき、これを利用して GPS 衛星の探索を行うため、体験教室等の開始直前にテストしておくと、児童がアプリを起動してから、正常動作するまでの時間が短い。新しい携帯端末であってまだ GPS 機能を使用したことが無い場合や、長い間 GPS 機能を使用しなかった場合には、GPS 機能が立ち上がるのに長い時間がかかる。表 2-2-1 に、計測例を示す。

GPS 機能を使用し続けている間は、ほぼ同じ衛星（最低４）からの電波を使用して位置を算出するため、誤差は一定である。従って、上記の「足による補正」の機能は有効である。しかし、一度アプリを終了して GPS を停止し、再び起動すると、前回とは大きく異なる位置が計測される場合がある。

表 2-2-1 新宿百人町における GPS 電波取得に要した時間の計測結果（2015.9.19）

メーカー	製品名	OS	No	GPS 待(秒)	機材番号	緯度	（分）※	経度	（分）
SONY	XPERIA Z2	4.4	1	37	ST2-12	36	5.152	140	5.808
	(SGP512JP/B)		2	30	ST2-13	36	5.146	140	5.806
			3	89	ST2-14	36	5.147	140	5.802
			4	4	ST2-15	36	5.146	140	5.802
			5	29	ST2-16	36	5.144	140	5.809
SONY	XPERIA Z	4.1	1	62	ST3-19	35	42.354	139	41.755
	(SGP312JP/B)		2	31	ST3-20	35	42.353	139	41.754
			3	61	ST3-21	35	42.355	139	41.76
			4	38	ST3-22	35	42.357	139	41.756
			5	23	ST3-23	35	42.351	139	41.756
ASUS	NEXUS7	4.4	1	62	AN2-27	36	5.142	140	5.801
	(2013)		2	16	AN2-28	36	5.146	140	5.807
			3	64	AN2-29	36	5.149	140	5.809
			4	23	AN2-30	36	5.147	140	5.806
			5	100	AN2-31	36	5.145	140	5.804

※緯度の分以下を示し、１分以下は、秒を使用せず分の小数として示す。

（６）データの原点の確認

後述のように、データ作成段階で、建物等を記述する座標系の原点の緯度・経度・標高をいくつかの方法で定義することができる。この数値が正しい値でなければ、背景との合成が正しい位置に行われない。奥尻島の例では、2013 年 6 月に、２台の携帯端末を用いて GPS 計測した、ランドマークである「徳洋記念碑」の緯度経度を用いてデータを作成し、2014 年 3 月に建物単体のデータを用いて表示位置の確認を行った（写真 3-1)。この結果に基づいて、より位置精度の高い機種を精選した上で、同年 10 月に体験教室を実施した。この時、セッション直前のテストで東西に約５ｍ程度の位置ずれが生じたため、同じ位置で緯度経度を計測し直し、正しい位置に表示できるように補正した。GPS により計測される緯度経度は、大気の状態などにより影響され、また稀に地震・地滑り等による大地の移動の影響も受ける。このため、古い計測値を用いると誤差が生じる。しかし、体験教室等の実施中に大きく変化することはないため、直前に測り直してデータを補正することにより、前述の「足によるキャリブレーション」により個々の機材で補正する手間を省き、円滑にセッションを実施することができる。

この直前の補正は、ModelIndex.txt に登録してある、個々のタブレットにセットアップされていたテキスト・エディタを用いて、モデルの座標原点の経度の値を、約５ｍに相当する 0.0005 度だけ増やす方法で補正した（東経 139.4500→139.4505)。

（7）記録データの回収・保存

セッション中にシャッターボタンが操作された場合には、携帯端末中に画像ファイルと、その時点での携帯端末の位置・姿勢・時刻を記録したデータが作成されている。これらを回収するためには、PC と USB 接続して、あるいは着脱可能な SD カード等を介して、必要なファイルを取り出す。

簡単には、終了時点での VirtualConverter ディレクトリ以下を携帯端末のファイル・マネージャの機能を用いて ZIP 圧縮し、このファイルを回収する。

２－３．データ作成方法

データ作成は、建物や町並を記録する任意形式のデータを作成し、これを解読するためのメタファイルを添付し、これらに目録を付けて、パッケージ化するまでの工程である。この内、メタファイルを作成する工程は一種のプログラミングであるため、２－４として別項を設けて解説した。

なお、本節の解説は、現時点で最も利活用の有用性が高い、現場での表示（VC-3M）のためのデータ作成を中心に記述しているが、数百年後の利活用を目的とした長期保存を目的としたデータ作成とも、多くの部分は共通している。保存すべきデータは他にも様々な方法によって作成済のものを扱う場面も想定される。その場合、メタファイルを作成する工程が最も本質的な作業となる。

（1）建物の三次元データの作成

建物の三次元データを作成するためには、以下のような方法がある。

① CAD 入力

1990 年代から、CAD(Computer Aided Design)ソフトウェアを用いて三次元データを作成する方法が実用化し普及した。その後、機械分野では数値制御式(NC)加工機を用いた CAM(Computer Aided Manufacturing)と連携することにより急速に普及した。建築分野でも三次元 CAD の導入自体は早かったが、急速に普及したとは言えず、二次元の CAD データを紙に出力した図面が引き続き広く使われている。近年では、BIM として、三次元データに様々な属性データを持たせて建物を表現する手法が普及し、標準的な IFC 形式が保存データのフォーマットとして勧奨されている。

図 2-3-1 は、福岡県住宅供給公社が建替えを行った峰花台団地の設計図から、CAD ソフト Microstation により入力したデータである（平成 8(1996)年）。当時は、CAD ソフト毎の独自フォーマットで保存されたデータを、コンバータにより、建設省版・景観シミュレータの独自フォーマットである LSS-G 形式に変換して、現場での説明等に利用した。

図 2-3-2 は、1980 年頃に、解体処理に先だって実測された研究施設の記録図面から、Autocad により入力したデータである(平成 23(2015)年)。データは後述の IFC 形式で保存し、この形式を解読するためのメタファイル IFC.cmm を添付したものをタブレット端末に搭載し、VC-3M アプリケーションを用いて、リアルタイム写真合成を行い、現在は高層集

合住宅や防災公園となっている現場での閲覧に使用している。

この 20 年ほどの間、ソフトウェアの性能自体は本質的に変わりないが、使用している PC の性能（メモリ容量、計算速度、３D 表示速度）は飛躍的に向上しており、大規模なデータを表示できるようになった。

図 2-3-1 峰花台団地の景観シミュレーションと、竣工後の比較(1996)

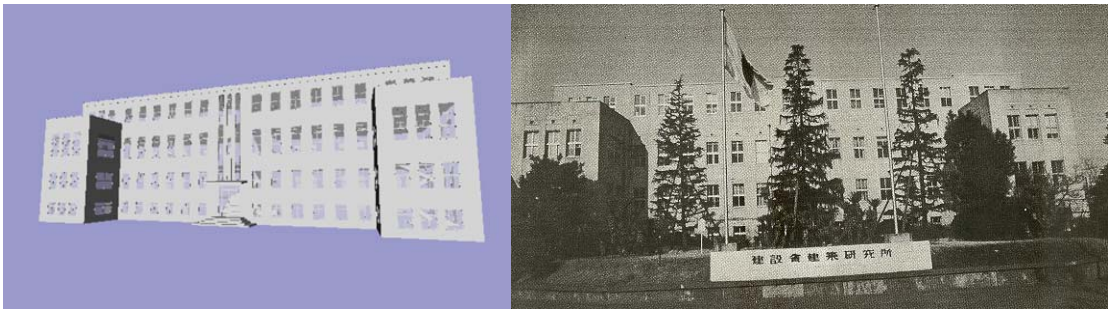


図 2-3-2 実測図からモデリングした旧建設省建築研究所の本館と、古写真(1976 年頃)

② 写真からのモデリング

立体的な形状を計測するために、ステレオ写真を使用する方法は、空中写真から等高線を抽出する方法を、地上写真にも適用することが一部に行われていた。1990 年代にその画像解析技術の自動化が進んだ。

建築物に関しては、直角、水平、垂直が仮定できる要素が多く含まれることから、１枚

の写真を基に立体的な形状を復原することができる。具体的には、写真上から二つの消点を抽出することができれば、建物の形状を求めることができる。このため、殆どの場合ステレオではない記録写真からでも、建物を復原することができる。このような作業を行うためのソフトウェアはいくつか存在している。

更に、一つの区域に関して多くの写真が残されている場合、写真そのものを変形して接合することは容易でないが、立体的に復原した断片を正しい位置に配置することにより、立体的な町並の復原が可能である。

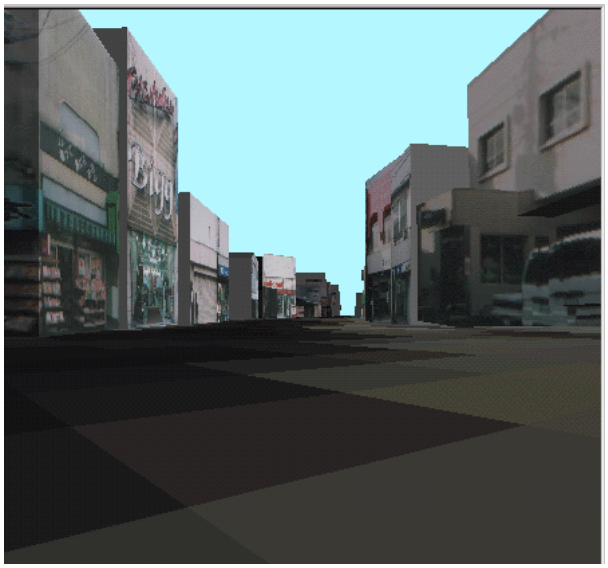


図 2-3-3 デジタルカメラ画像からモデリングした建物を配列した町並（幕張 1996 年）

幕張駅周辺においてデジタルカメラで撮影した沿道の現況建物から復原した建物を配列して作成した町並のデータ(平成 8(1996)年、旧建設省建築研究所)を図 2-3-3 に示す。図 2-3-4 は、奥尻島において昭和 58(1983)年の日本海中部地震による津波の直後に撮影された調査写真（サービス判印画紙）から国総研が平成 14(2012)年に復原した当時の住宅である。

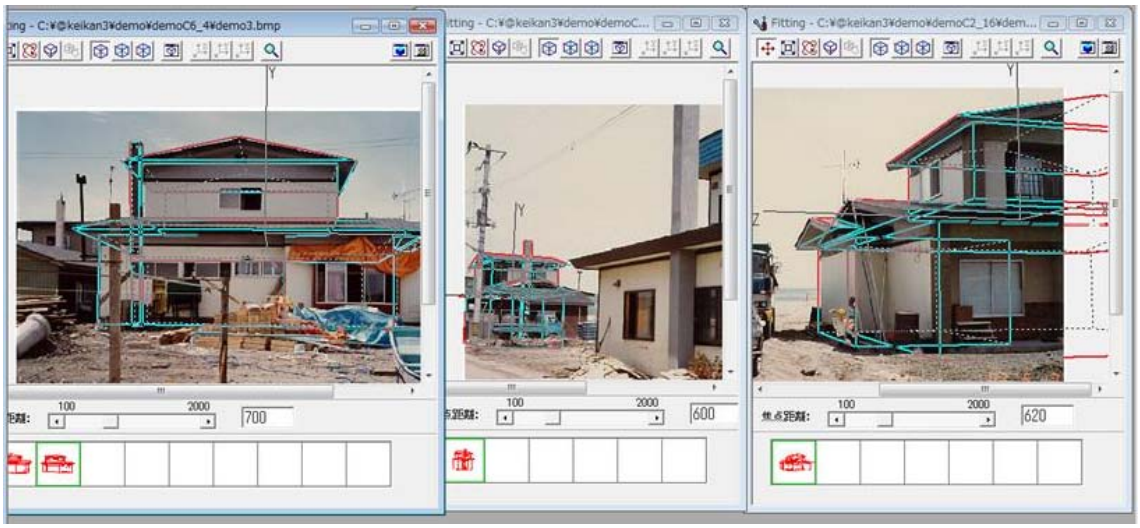


図 2-3-4 1980 年代の古写真から復原した奥尻島青苗の岬地区の住宅

③ レーザー計測

レーザー計測技術により、立体的な地形や建物を点群データとして計測することができる。2000 年頃から、航空機から地形を計測する技術が普及し、DEM データとして利用されるようになった。更に 2010 年代に入ってから、車載型のレーザー計測装置（MMS）が開発され、沿道の町並の形状を能率的に計測できるようになった。図 2-3-5 はつくば市内の建築研究所の実験施設を国総研が計測した点群データである(2013 年計測)。点群データは、XYZ 座標値を羅列しただけの単純なファイル形式による大規模なファイルである。これを建物単位、壁単位、等に分節化し、更には使いやすく小さいサーフェスモデルに変換する処理については、国総研の別課題において研究された(2016 年)。保存時点で適切な解析処理がすでに行われていれば、CAD 入力されたデータ等と同じ方法で扱うことができる。

本研究においては、点群データそのものを保存する場合に際して、位置座標のオフセット等を反映させた上で、レーザー計測装置の回転スキャン動作の中で取得された、連続する一群の計測点を折れ線として集約するところまでを行った。

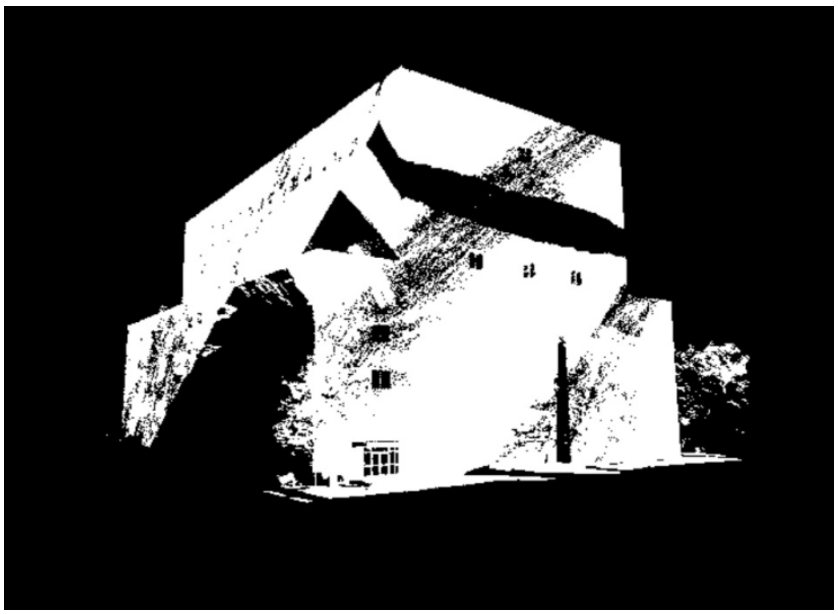


図 2-3-5 MMS で取得した建築研究所の実験棟建物の点群データ(実大構造物実験棟,2013)

（２）地形データの作成

① DEM データの利用

ある区域をメッシュに分割し、それぞれのメッシュの標高の平均値を記録したメッシュデータは、DEM(Digital Elevation Model)として、1990 年代後半から普及した。国土地理院ではこれを受け、数値地図標高として、(一財)日本地図センターからの有償販売を開始し、2010 年代には、「基盤地図情報」として無償で利用できるサービスを開始している。

様々な GIS ソフトで DEM データを採り込んで利用することができる。旧建設省建築研究所も、景観シミュレータ Ver.2.03(1997)*11 からコンバータを提供している。当時は統一された規格がまだなく、個別に作成された DEM データには様々な形式のヴァリエーション

が存在しており、試行錯誤しながら正しい変換結果を得るような作業を行う必要があった。

DEM に対応したオルソ画像が利用できる場合には、これをテクスチャとして地形に適用することにより、樹木・裸地などの識別が可能なデータを作成することができる。例えば景観シミュレータにおいては、地形メッシュが長方形領域である場合には、テクスチャ編集画面で横倍率、縦倍率をそれぞれ領域の東西長さ(m)、南北長さ(m)に指定すればよい。テクスチャ編集地形データが十分に精密である場合には、テクスチャではなく、頂点カラーとして定義する方法も可能である。国土交通省版・景観シミュレーション・システムに付属した「貿易コンバータ」で、DEM を LSS-G 形式に変換する設定画面において、頂点カラーを使用するチェックを入れた上で、使用する空中写真等の画像ファイル名を指定することで、頂点カラーのついた地形等のデータを作成することができる。多面体として表現された地形を構成する頂点位置を地表面に投影した XY 座標に対応する位置の色彩を、指定された画像ファイルのメッシュから補完計算を行った上で抽出する処理を行っている。

② 紙地図の等高線からの生成

古い都市計画図等には、造成後の現在とは異なる従前の地形が等高線と単点で表現されている。三次元モデルを作成するために、大判のスキャナを用いて紙地図から画像ファイル（本件では tiff 形式）を作成し、これを画面表示した上で、マウス操作で等高線を抽出するような作業が必要となる。奥尻島の平成 5 (1993)年の基本図（1:2500）から作成した地形データとその作業過程を、図 2-3-6 に示した。

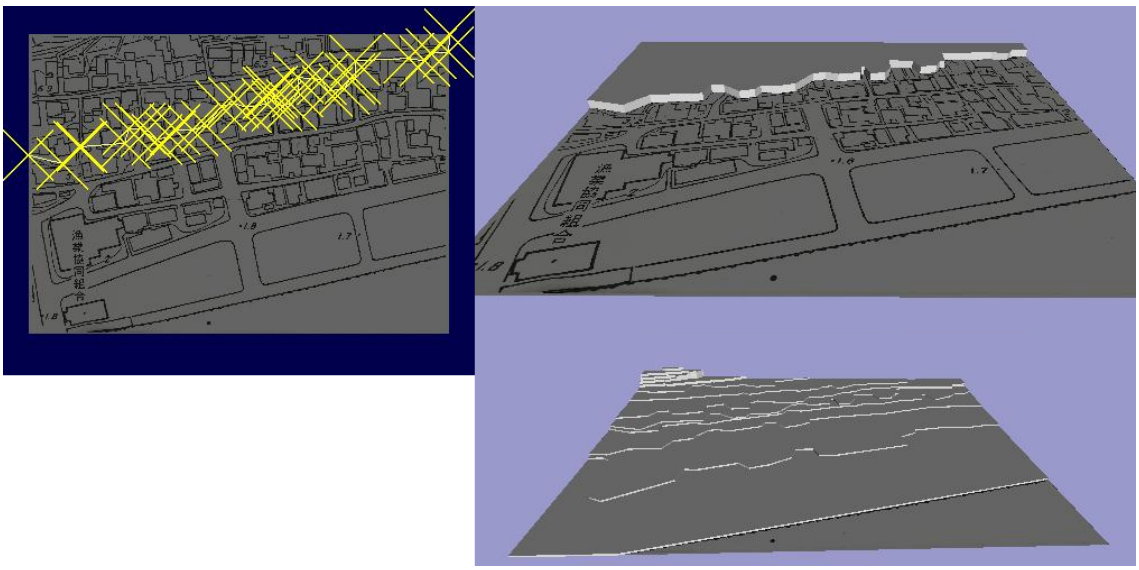


図 2-3-6 古い基本図(1:2500,1993)から等高線を抽出し、地形を再現する

なお、マウスとグラフィックディスプレイが普及する前の 1990 年頃までは、大判のデジタイザにテープなどで地図を固定し、磁気センサとボタンを有するポインティングデバイスを図上の点に合わせてボタンを押し、座標値を計測・入力する方法が広く行われていた。

③ ステレオ空中写真（ライブラリ）からの復原

ステレオ空中写真がアーカイブされていれば、これを解析して三次元データを作成する

ことができる。この場合、地形だけではなく、建物も併せて取得できるため便利である。

2001 年度に実施した「まちづくりのためのコミュニケーション・システムの開発」においては、参加した地方公共団体が 1:2500 基本図の作成のために過去に撮影した空中写真（多くは測量会社が保管）、あるいは国土地理院が 1:5000 国土基本図の作成のために過去に撮影した空中写真から、地形と建物をモデリングし、これを下図として計画案のデータを作成した。

図 2-3-7 ステレオ空中写真から復原した地形（広島,2001）

④ 衛星画像からの復原

国総研が 2004~6 年にインドネシア人間居住研究所の協力を得て、同国内の計画的な住宅地の排出量実態調査と、今後排出量を増加させない計画の検討を行った際には、日本の衛星「だいち」が撮影したステレオ衛星画像を使用した。同国においては、軍事的理由から空中写真の国外への持ち出しは禁止されている。従って、90 年代までは地形データを作成するためには、現地で空中写真の解析を行うか、既存地形図に描かれた等高線から旧来の手法で立体的な地形を作成する必要があった。衛星画像にはこのような制約がないため、地球環境研究に広く利用されている。「だいち」のリニアセンサーは、衛星から地表を見下ろす際に、真下だけではなく、前方と後方の斜め画像も取得する。これにより、あるエリアに関して、衛星が通過する前に撮影した画像と通過した後に撮影した画像をステレオペアとして解析して、既存の市街地と周辺地形の三次元データを作成することができる。

この方法をバンドン市とチレボン市に適用した[29]。このうちバンドン市周辺のデータの例を図 2-3-8～9 に示す。図 2-3-8 は、同じエリアに関して、ステレオ衛星画像から解析した地形と、地形図から抽出した等高線に基づき従来方法で作成した地形の細密度を比較したデータである。図 2-3-9 は、地形図から作成した概略の広域地形を遠景とし、計画団地の

周辺の地形についてはステレオ衛星画像から作成した詳細地形に衛星画像をテクスチャとして貼り付け、更にその上に集合住宅団地と戸建住宅団地計画案の CAD データを合成して作成したデータである。

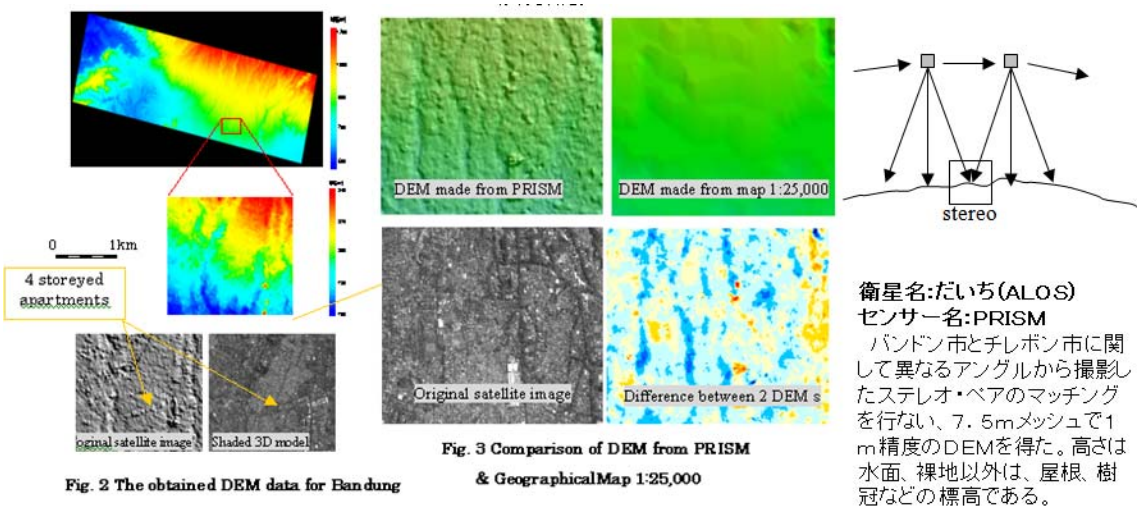


Fig. 2 The obtained DEM data for Bandung

Fig. 3 Comparison of DEM from PRISM & GeographicalMap 1:25,000

図 2-3-8 ステレオ衛星から復原したバンドン市内の地形と、従来データの比較

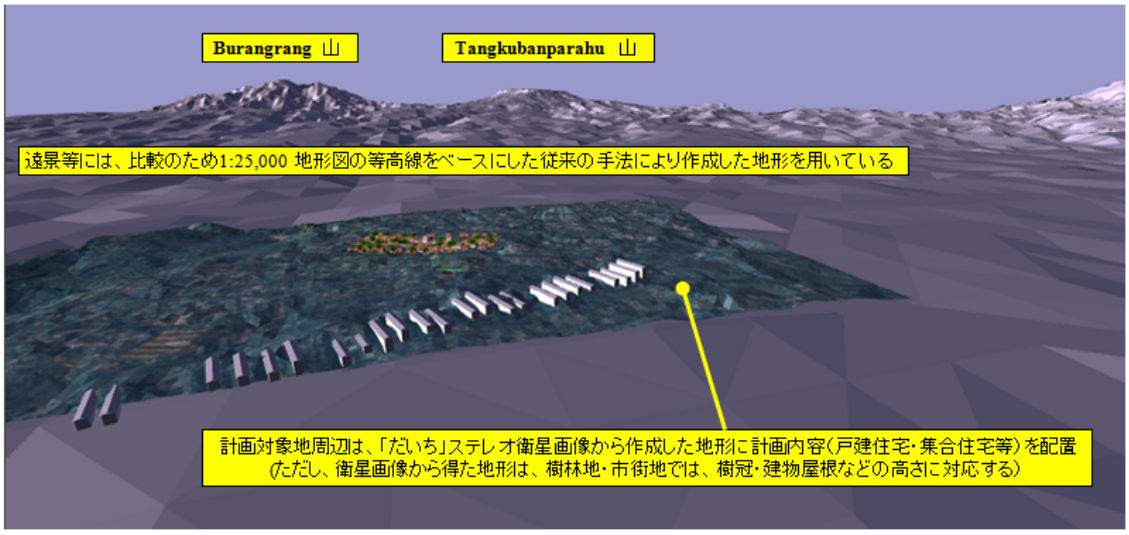


図 2-3-9 地形図等高線、ステレオ衛星画像解析結果、CAD 入力による計画案の合成

⑤ 基盤地図情報の利用

基盤地図情報（標高）は、一般地域においては、1:25,000 地図の等高線から補完して作成した１０ｍのメッシュデータであり、擁壁等の周辺で実際とはかなり異なる場合がある。一方、レーザー計測した結果は５ｍメッシュで作成されており、より精度が高い。1:2,500 地図に対応する精度である。

（３）団地の地盤面の作成

団地の地盤面が整地される前の地形データしか得られない場合には、地形の編集を行う必要がある。

① 景観シミュレータの高台整地機能の使用

前項のような各種の方法で作成した地形データに対して、ある一定の平面的なエリアを平坦に整地すると共に、周辺の地形との間に指定した勾配の法面（切土・盛土）を追加して隙間ない閉じた空間図形を生成する機能を、景観シミュレータのためのプラグインとして作成した。国総研のホームページからの公開プログラムに含まれている。この処理は、多角形として入力された地盤面から、指定した勾配で十分の法面テンプレートをまず作成した上で、これと地形との間の図形演算を行う処理を内部で行っている。

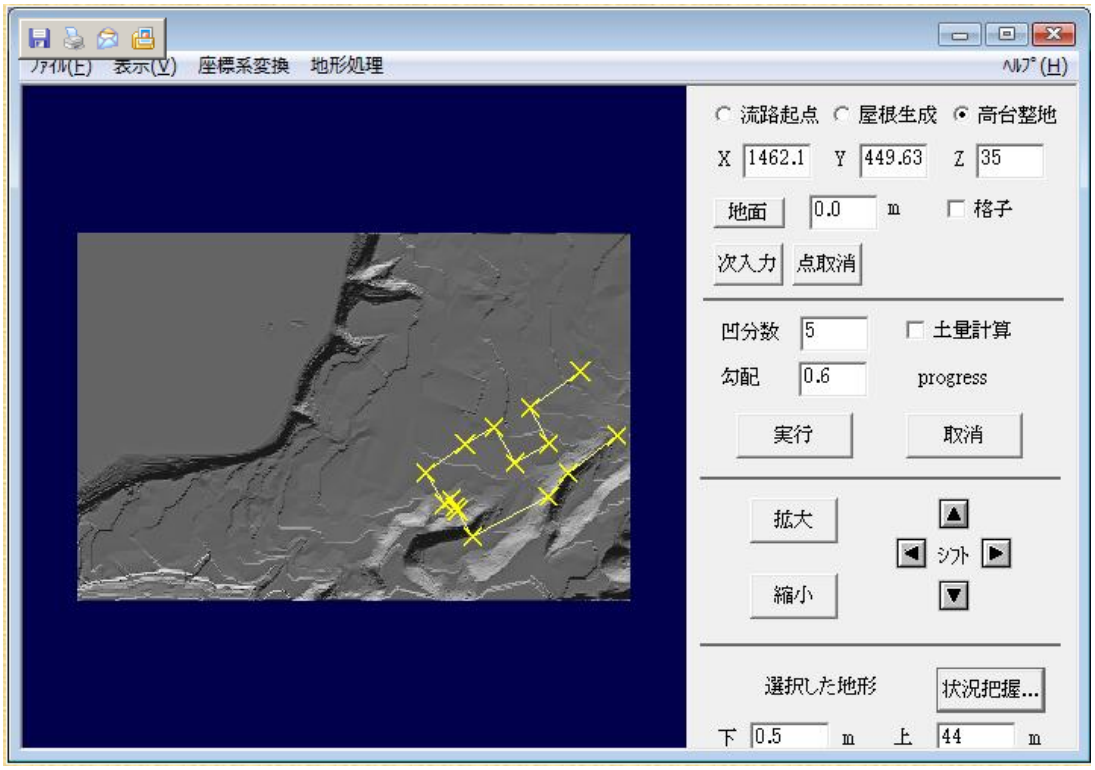


図 2-3-10 復原した地形の上に、造成エリアと標高を設定する

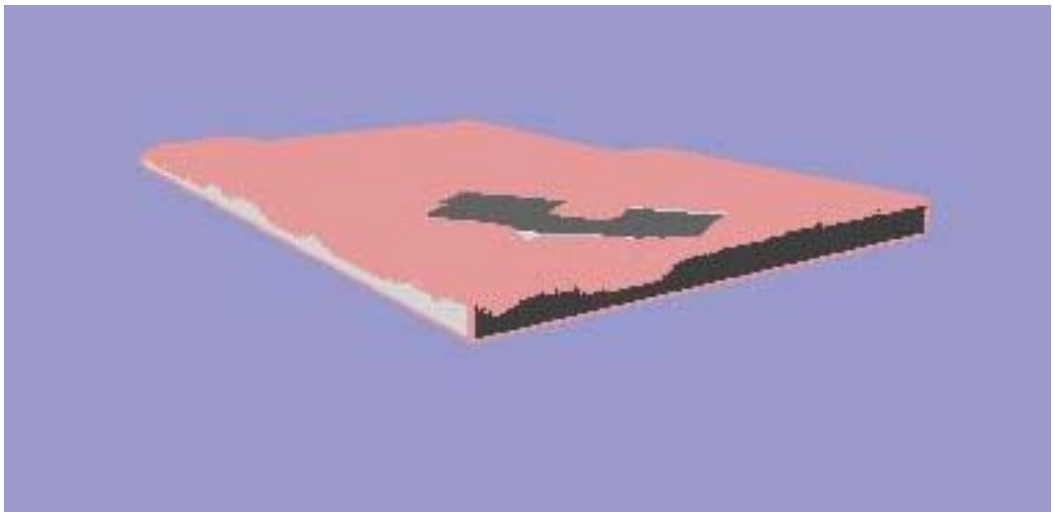


図 2-3-11 高台整地の図形演算結果

② 細部の編集

整地面、法面等の仕上げを、カラー、テクスチャ等を用いて編集する。

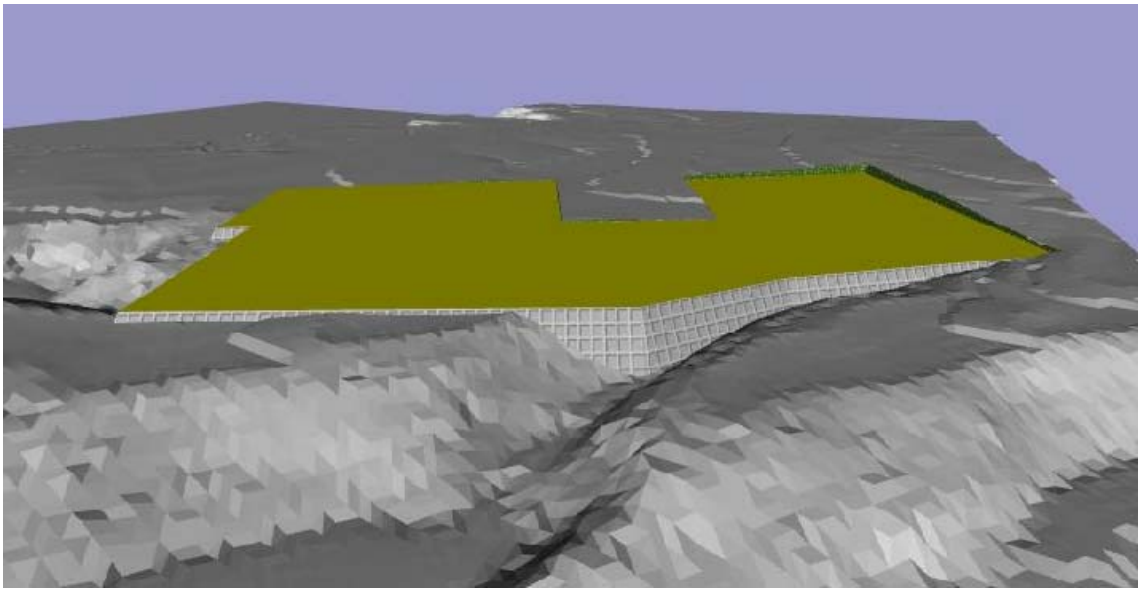


図 2-3-12 法面にコンクリート、植栽等を適用した。

（4）町並データの作成

① 建物の配置

原地形データの上に、被災前の空中写真をテクスチャとして貼り込んだデータを下地として、この上に個別にモデリングした道路と、古写真から個別に復元した住宅をそれぞれの位置に配置した結果を図 2-3-13 に示す。

図 2-3-13 古写真からモデリングした住宅を配置して復原した町並

② 属性データの付与

個々の住宅に、写真から判読できる階数、構造、屋根材料と共に、復原の根拠となった古写真の ID 等も属性として設定することにより、復原町並の三次元データを、資料整理・検索の手掛かりとしても利用できるようになる。

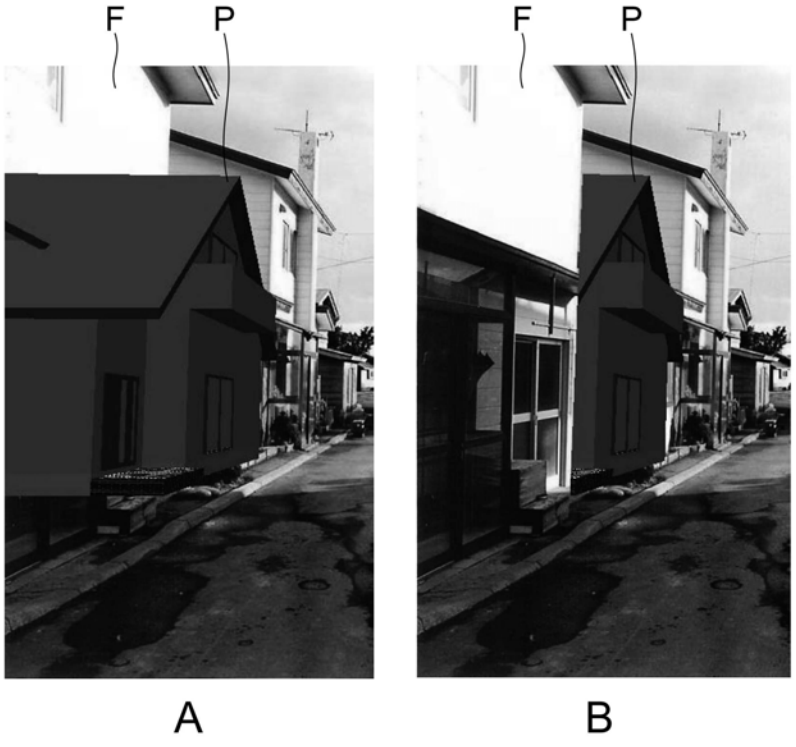
③ 地盤面の指定

GPS センサで計測した現在位置の直下に地盤の属性(&GROUND)が設定されている面が存在する場合には、表示に使用する高さを地盤面＋視点高さ 1.5m とする。これにより、造成等が行われた場所において、過去の地盤面に存在していた町並を地下に表示したり、現在の地盤と同じ高さに表示したり選択できるようにしている。但し、津波高さの水面を追加して建物の一部が水没しているように表現するなど、細部の工夫を行わなければ、単に縮小模型のような町並が足元に置かれているようにしか見えない。

④ 隠蔽面の作成

町並の中の空地に建物を表示する場合で、両隣の建物が現存するような場合には、両隣の建物の裏側に隠れる部分を表示しないようにする処理を行う必要がある。このためには、現存する両隣の建物の外壁を透明な板として作成し、復原建物に加えておく必要がある。

A では既存建物 F に復元建物 P が上書きされるため、手前にあるように見える。



B では復元建物 P に F の側面を透明な板として加えてあり、その背後となる P の部分が消える。

図 2-3-14 復原建物が隣の現存建物に隠れるような処理

（５）テクスチャファイルの作成

写真から立体的な復原を行った壁面等に、対応する元の写真の画像をテクスチャとして貼り込むことによって、再現性を高めることができる。奥尻島の住宅の復原作業に用いた

リアル・モデラーは、個々の建物の構成面を含む小さな画像を原写真から別ファイルとして切り出し、テクスチャとしてそれぞれの面に適用するような処理を行っている。

（６）ローカル座標系の計測と登録

① 座標原点の緯度・経度・標高の計測

町並のデータの座標原点の緯度経度標高を正しく指定することにより、携帯端末を用いて、正しい位置に表示することができる。図上計測は、例えば国土地理院のホームページから地図を検索すると、参考情報として図上の点の緯度経度を知ることができる。基盤地図情報は、緯度経度をベースとして作成されているため、補間により座標原点の緯度経度を把握することができる。また可能な場合には、閲覧に使用する携帯端末を用いて現場でGPS を用いて直接計測することも可能である。VC-3M を用いてシャッターを操作すると、たとえ正しい位置に表示が行われていない場合であっても、mobile.ja.scn に、シャッター時点で計測された GPS 座標値が記録されているので、これを用いてデータを準備し、精度を高めることができる。

計測された座標原点の緯度・経度・標高を表示コンテンツの中に入力するためには、３つの方法を用意した。第一は、「見たい場所」の一覧を表示するためのリスト(ModelIndex.txt)の中で指定する。第二は、メタファイルの中で指定する。第三は、データファイルの中で指定しておき、これをメタファイルで読み出して指定する。

② メタファイルへの登録

表示すべき建物等の三次元データの記録形式を解読するためのメタファイル（形式定義ファイル）を準備して、データファイルとのペアで、一覧ファイルに登録することにより、任意のデータ形式の記録を現場で再生することができる。

メタファイルには、データファイルを解読する方法を記述する以外に、固定的な形状を記述することもできる。GPS により計測される高さの精度が低い場合、あるいは現在の地盤面と異なる、嵩上げ前の過去の地盤面を用いて閲覧したい場合、あるいは手前にある現存建物に隠れる部分を消去して表示したい場合等に付加する情報は、データファイルの中に含めることもできるが、例えば隣接する建物は変化する可能性があるため、表示段階でのみ使用し、長期保存データには含めない方が良い場合もある。このような場合には、表示段階で使用するメタファイルの中に隣接建物の外形を固定的な形状として記述する方法が望ましい。同様に、記録された過去の町並とは異なる現在の地盤面から見下ろした風景を表示したい場合にも、メタファイルの中で現在の地盤面を記述する方法は有用である。

メタファイルの作成方法に関しては、２－４で解説する。

③ 表示選択用リスト・ファイルへの登録

わかりやすい見出し名、データファイル名、メタファイル名、および座標原点の緯度経度標高の計測値をリスト(ModelIndex.txt)に登録することにより、VC-3M で一覧表示の中から選択し、表示することができるようになる。この登録方法については、２－５で解説する。

２－４．メタファイルの作成

本節では、保存データに添付するメタファイルの作成方法を解説する。

（1）考え方

メタファイルは、一種のプログラム言語ないしスクリプト言語であり、以下の特徴をもつ。

① C言語に準拠した文法体系

・プログラムの中で変数および関数を定義して処理を記述する。
・プログラムの記述は、「;」により終了する。
・/*　*/によりコメントを挿入することができる。
・//によりコメント行を挿入することができる。
・main 関数から処理を開始する。
・if, switch, for, do, while 等により条件分岐、ループを記述する。
・if 文を簡略化した複合式「if(式１)?(式２):(式３)＞」の記法は使用しない。
・変数と一次元配列を使用する。
・exit 文により任意の場所で処理を中断・強制終了することができる。

一方で、以下の機能は省略されている

・fopen 関数を持たない。入力するデータファイルは最初から開かれている。
・入力するデータファイルには、固定長配列のようにアクセスすることができる。
・fscanf 関数は使用せず、代わりに scanf 関数（後述）によりデータファイルから読み込む。
・配列は１次元だけである
・構造体を定義し使用することはできない。固定長配列のみを使用する。
・malloc 関数等の動的メモリ管理を行わない。
（解読を行うデータがあらかじめ定められていることから、処理に必要な配列の長さは既知であり、大域変数として宣言する十分な長さの固定長配列により処理可能である）
・ポインタを使用することができない。
（セキュリティを高めるための配慮である）
・#define, #if, #include などのマクロは使用できない。
（「#」で始まる行は、コメントとして解釈される）
・リンカはなく、全ての処理を１本のメタファイルの中で記述する。

他方で、以下の機能が追加されている

・main 関数を省略する記述が可能である
（関数定義を行わない場合）
・宣言されない変数への代入が行われた場合、そのステップで宣言されたものとみなす
・scanf 関数の仕様が異なる
（代入する変数は１個のみとし、戻り値はパターンマッチの結果である。失敗した場合には、ポインタは動かない）

② LSS-G 形式のコマンドを部分集合として含むライブラリ関数

・LSS-G 形式のファイルを構成するコマンドは、C 言語の関数の書法である

変数名＝コマンド（引数）；　または

コマンド（引数１, 引数 2, ・・・）; という記述を基本とする

・LSS-G 形式では、例えば座標値を表す引数は数値（定数）であるが、ライブラリ関数においては、変数などから構成される数式を用いることができるように拡張した。

・LSS-G 形式では、引数の数を無制限とする記述を可としており、継承した。

・LSS-G 形式では、引数を省略する記述を可としており、継承した。

・メタファイルによる変換処理の記述を容易とするために、面の頂点列の末尾に頂点を追加する FACE_VERTEX コマンドを新たに追加した。

・厖大に蓄積されている LSS-G 形式のファイルを、記述している形状が既知でバグの無いメタファイルとしてコンパイル・実行することができ、これを用いて表示やデータベース処理などの利活用処理系の開発、テスト、デバッグを行うことができる。

③ OS に依存しない処理系の上でのコンパイルと実行

・コンパイラ、インタープリタのソースコードは C 言語で記述されているため、C コンパイラを幅広いプラットフォームに移植することができ、そこでメタファイルをコンパイルしてデータファイルを読み込む処理を実行することができる。

（２）メタファイルの種類

メタファイルは次のレベルのものが作成可能である

① 固定形状

データファイルを参照しない固定形状をメタファイルだけで記述することもできる。具体的には、FILE 関数しない固定的な（パラメトリックではない）LSS-G 形式のファイルは、固定形状のメタファイルとして処理できる。任意形式のファイルを入力するメタファイルとしては無意味であるが、様々な処理系のテストデータを簡単に用意することができる。長期保存されたデータの解読結果を利活用する新たな処理系の開発にあたって、初期の段階でのテストとデバッグのために、既存の LSS-G 形式のデータをサンプルとして用いる方法は、極めて有用である。

② パラメトリックな形状

小数のパラメータからパラメトリックな部品の生成をメタファイルで記述できる。景観シミュレータのために作成された外部関数のソースコードを修正（簡略化）することにより、関数としてメタファイルのメイン関数から呼び出すことができる。このとき、データファイルとはパラメータのみを定義する小さなファイルである。

③ 高度な文法形式をもつデータ形式の解読処理

景観シミュレータの外部関数として実装されているファイルコンバータが利用可能な場合には、ソースコードを修正することにより、当該データ形式のためのメタファイルを作成することができ、本処理系を用いて Windows 以外の環境においても利用することができ

る。重要な特徴は、このようにして作成したデータファイルとメタファイルの組は、景観シミュレータから独立して、未来の様々な処理系で利活用することができる、という点である。このことについては、利活用システムの開発を解説した第３章で詳しく解説する。

CAD や GIS アプリケーションを使用して入力された多くの三次元データは、高度な文法形式を有するデータ形式で保存されている。保存データに添付するメタファイル作成に際して、当該データ形式の全ての仕様に対応する必要はなく、あくまで保存データの中で参照されているキーワード等に対して対処できれば十分である。このため、メタファイルの中で解析に用いる固定長配列等も、必要最小限のサイズとすれば十分である。

（３）保存データの基本的な構造の把握

① 文字コード

２バイト系の文字に関して、様々な表記法が存在する。例えば、データの中に直接埋め込まれている場合（例 LandXML）と、ascii 文字を用いて間接的にコーディングされている場合がある（例 IFC）。

データの中に直接２バイト系コードが埋め込まれている場合には、ヘッダー部分に文字コードが表記されている場合がある（例えば XML 系）。しかし、例えば「メモ帳」エディターを用いて、ヘッダー部分を保ちながら異なる文字コードに簡単に変換することができる。従って、保存に際しては、ヘッダー部分の表記が正しいかを確認しておく必要がある。

利活用に用いるシステムで使用する文字コードが、原データとは異なる場合で、２バイト系文字コードを利活用システムでメッセージの表示などに使用したい場合、コード変換を行う必要がある。

コンパイラ処理系においては、コンパイラのソースコードに定数で文字コードを埋め込んでおき、これをバイナリで判定することにより、利活用処理系の文字コードを判別する方法を採っている。しかし、これは、メタファイルやデータファイルの文字コードと必ずしも一致することを意味するわけではない。例えば、Windows 上の開発環境である VS2005 の初期設定においては、utf-8 形式で保存されたソースコード中に埋め込まれたリテラル定数であっても、fprintf 関数等でファイルに出力した場合、Shift-JIS コードが出力される。

あるファイルの文字コードを明らかにするために、冒頭部分に、既知の２バイト系文字列を埋め込んでおく方法が有効である。ヘッダー部分に使用する文字コードを記載する方法がしばしば用いられているが、データファイルを編集・再保存が行われると、ヘッダー部分がそのままで、データ内部の２バイト系の文字列のコードが変換されているような場合があるので注意を要する。データファイルを保存処理する場合には、冒頭部分に記載された文字コードで保存するのが親切である。メタファイルの文字コードは、上記のように日本語などの２バイト系の文字コードを含むコメントやメッセージ出力が記述に含まれる場合に意味をもつ。処理の最初に logf(“仮想コンバータ”); といったメッセージ出力をメタファイルの処理の冒頭に含めておくことが、以後の保存処理と、後世の利活用にとって有効である。

② ブロック構成

多くのフォーマットにおいては、データ全体はヘッダー部分といくつかのデータブロックにより構成されている。この構造を把握し、例えばヘッダーを解読する header 関数と、データ本体を解読する main 関数から、header 関数と必要回数だけ data 関数を呼び出すようにメタファイルを構成するのがわかりやすい。

③ コマンドとパラメータの抽出

各種フォーマットにおいては、多くのキーワードやコマンドが定義されているが、実際の保存すべきデータにおいては、その一部しか使用されていない場合が多い。使用されていないキーワードを処理する関数を削除したメタファイルであっても、添付保存するメタファイルとしては十分である。

利用頻度が高いデータ形式の場合には、仕様書・定義書に定められたコマンド群の全てに対応した共通の親メタファイルから出発し、添付ファイルを作成する段階で、参照されない関数等を削除するような方法が能率的である。

本処理系においては、scanf(“文字列)”の形式でキーワードやコマンドを識別する。標準 C 言語の fscanf 関数においては、値を代入する変数がない場合には、この文字列のパターンが一致するところまでポインタが進み、戻り値は成否にかかわらず一定である。本処理系の scanf 関数においては、パターンが完全に一致した場合のみポインタが進み戻り値 1 を返し、そうでなければポインタは不変で戻り値 0 を返す。これにより、キーワードを抽出処理における gets 関数等を用いたバッファリングを省略している。

④ シンボル・テーブル

高度なデータ形式においては、シンボル（例えば繰り返し使用される図形の名称）を定義し、それを後の処理の中で参照することが行われる。このため、通常の場合にはシンボル・テーブルを作成する処理が行われる。

本処理系の場合には、解読対象とするデータファイルが固定されていることから、シンボルが最初に検出された位置の先頭アドレス（整数値）をもって、あるシンボルを特定することにより、長さが無制限のシンボルを、char[]配列を用いたバッファリングや、malloc/free 関数を用いたメモリ管理を行うことなく処理している。

（4）プログラミングとデバッグ

現在までに試作した 4 種類の実装形態に共通して、データファイルの利活用段階における解読処理は、メタファイルのコンパイル処理と、生成したコンバータの実行段階からなる。よって、エラーはコンパイル・エラーと、実行エラーに分けられる。コンパイル・エラーが生じた場合には、実行段階に移行することなく処理が終了する。

テキスト・エディタを用いて、メタファイルを編集し、データファイルと共にシステムに読み込むことにより、テスト･デバッグを行うことができる。VC-3M の場合には、Android 上で動作するテキスト・エディタにより現場で編集・修正を行い、直ちにコンパイルと実行（読み込み→表示）を行うこともできる。

保存データを現場での再生閲覧するために使用する処理系は、なるべくシンプルで軽く、バグの少ないものが望ましいことから、現場での閲覧のための処理系に多くのデバッグ機能を追加する計画はない。現時点では、デスクトップ PC の上で動作する VC-1C または VC-2V を用いて、メタファイル完成直前までのメタファイルのプログラムとデバッグ作業を行うのが能率的である。

① コンパイル時のエラーメッセージ

コンパイル・エラーを、表 2-4-1 に一覧する。エラー処理は、仮想コンバータに含まれる err_ss 関数(cci_misc.c)を通じて出力される。この関数は、二つの文字列（エラーメッセージ）を引数として取得し、エラー内容と、デバッグの参考となる文字列を出力する。また、二つ目の文字列を空とする err_s 関数、二つ目の文字列を数値とする err_fi 関数もエラーメッセージ出力に用いている。

エラーの原因となったメタファイル文法に関しては資料４－１に、またライブラリ関数の使用方法の詳細については、資料４－２に解説する。更に、新たなデータ形式に対するメタファイルの作成のための参考として、資料４－６に各種三次元データ形式とメタファイル作成例を解説する。

表 2-4-1 コンパイル・エラー　メッセージ一覧

(err_ss 関数が出力するメッセージ)
未定義の二項演算子(Int) (code:132)
未定義の二項演算子(Float) (code:150)
未定義の演算子(Quat) (code:171)
未定義の関数 (code:319)
scanf 書式(書式文字列)(file:161,173,253,263)
ミスマッチ発生(文字列型書式文字列) (file:446)
// scanf コマンドで、文字列型入力を含む書式文字列に従った入力に失敗した場合, EOF 到達など
（がない(引数が ON または OFF のコマンド) (pars:123)
ON でも OFF でもない(引数が ON または OFF のコマンド) (pars:129)
）がない(引数が ON または OFF のコマンド) (pars:131)
；がない(引数が ON または OFF のコマンド) (pars:133)
構文エラー（解釈できないトークン） (pars:237)
識別子が重複している(宣言された関数名) (pars:274)
関数が再定義されている(宣言された関数名) (pars:319)
関数プロトタイプが不一致(宣言された関数名) (pars:323)
｝を検出(main()関数が無い形式) (pars:412)
ReadOnly モードで書き込み statement(コマンド) (pars:599,992)
//データベースから出力ファイルを生成する形式定義ファイルに、データベースへの書き込みコマンドを検出
未定義の関数を使用(要求された関数名) (pars:650)

不正な記述(文として解釈不能な記述) (pars:678,1177)
型が不適切(=代入) (pars:786)
未解決の数値型の組み合わせ FI (pars:850)
//異なる数値型への変換が解決できない場合
引数の数値型が不適(三角関数) (pars:879,888)
void 型関数が式の中で使われている(使われた関数名)(pars:975,1063)
不明な型の変数が参照された(変数名) (pars:1074)
関数の引数個数が不一致(関数名)(pars:1214)
argsV 引数に二重の＊あり token.text (pars:1248)
argsV 引数がシンボルでない token.text (pars:1253)
argsV 引数が変数でない token.text (pars:1257)
argsV 引数が整数型の変数でない token.text (pars:1260)
//変数名を引数とする関数における引数の指定方法にエラーがある場合
引数の型が不正(LINK_XFORM) (pars:1342)
第二引数｛LOAD,PRE,POST｝がない(LINK_XFORM)(pars:1355)
第二引数｛LOAD,PRE,POST｝が不正(LINK_XFORM)（pars:1372）
第三引数｛IDENTITY,TRANSLATE・・・｝が不足(LINK_XFORM)(pars:1377)
第三引数｛IDENTITY,TRANSLATE・・・｝が不正(LINK_XFORM) (pars:1397)
数値の数が不整合(LINK_XFORM) (pars:1408)
printf %n に対応する引数が変数でない(引数部分の記述)(pars:1478)
代入がない scanf の書式文字列が不正（書式文字列）(pars:1585)
scanf の書式文字列の次のトークンが不正(トークン) (pars:1589)
scanf 引数が変数でない(引数部分の記述)(pars:1606)
scanf 引数が整数変数でも浮動小数変数でも四元数でもない(引数部分の記述) (pars:1610)
scanf 型指定と代入する変数の型が異なる(書式文字列) (pars:1630)
四元数型引数が二以上(COORD) (pars:1782)
）が無い（COORD）(pars:1783)
引数がない（COORD）(pars:1789)
浮動小数型、整数型の引数の数が３に不足（COORD） (pars:1796)
浮動小数型、整数型の引数の数が３を超過（COORD） (pars:1797)
引数がない（NORMAL）(pars:1834)
引数の数が３に不足（NORMAL） (pars:1836)
引数の数が３を超過（NORMAL）(pars:1837)
引数がない（TCOORD）(pars:1890)
引数の数が２に不足（TCOORD） (pars:1892)
引数の数が２を超過（TCOORD） (pars:1893)

引数がない（COLOR）(pars:1920)
引数の数が３に不足（COLOR）(pars:1922)
引数の数が４を超過（COLOR）(pars:1923)
引数の数が２ではない（FACE_COLOR）(pars:2019)
引数の数が２ではない（FACE_TEXTURE）(pars:2023)
引数の数が２ではない（FACE_NORMAL）(pars:2027)
引数の数が２ではない（FACE_MATERIAL）(pars:2030)
引数の数が２ではない（LINE_COLOR）(pars:2040)
引数の数が２ではない（LINE_NORMAL）(pars:2045)
引数の数が２ではない（LINE_TEXTURE）(pars:2049)
引数の数が２ではない（LINE_MATERIAL）(pars:2053)
引数の数が２ではない（GROUP_MATERIAL）(pars:2057)
引数の数が２ではない（GROUP_TEXTURE）(pars:2061)
引数の数が２ではない（GROUP_ATTRIBUTE）(pars:2065)
引数の数が２ではない（LINK）(pars:2069)
引数の数が１ではない（TIME）(pars:2098,2101)
引数の数が１３ではない（CAMERA）(pars:2108)
引数の数が８ではない（LIGHT）(pars:2113)
引数の数が８を超過（LIGHT_GROUP）(pars:2118)
引数の数が１ではない（MODEL）(pars:2130)
引数が文字列型ではない (pars:2132)
引数の数が１ではない（IMAGE）(pars:2144)
引数が文字列型ではない（IMAGE）(pars:2146)
第一引数：種別が IKE ではない（EFFECT）(pars:2152)
第二引数がない（EFFECT(IKE)）(pars:2156)
緯度の数値はダブルクォーテーションマークで括る（EFFECT(IKE)）(pars:2158)
第三引数がない（EFFECT(IKE)）(pars:2161)
経度の数値はダブルクォーテーションマークで括る（EFFECT(IKE)）(pars:2163)
第四引数がない（EFFECT(IKE)）(pars:2166)
高さの数値はダブルクォーテーションマークで括る（EFFECT(IKE)）(pars:2168)
第五引数がない（EFFECT(IKE)）(pars:2171)
データファイル名メタファイル名はダブルクォーテーションマークで括る EFFECT(IKE) (pars:2173)
引数の数が６を超過（EFFECT(IKE)）(pars:2196)
引数の数が１ではない（EFFECT_GROUP）(pars:2201)
引数の数が７ではない（SCENE）(pars:2210)

引数がある LEN (pars:2216)
引数がある SIORI (pars:2220)
引数の数が 1 ではない SEEK (pars:2224)
引数が整数型ではない SEEK (pars:2225)
引数がある GETC (pars:2229)
引数がある GETS (pars:2233)
引数の数が 1 ではない GETINT (pars:2237)
引数の数が 1 ではない GETFLOAT (pars:2241)
文字列引数の数が 1 ではない（ _Q()） (pars:2252)
引数がない _Q() (pars:2271)
引数の数が 4 未満（ _Q()）(pars:2273)
引数の数が 4 を超過（ _Q()） (pars:2274)
引数の数が 1 ではない（四元数成分取得）(pars:2284)
引数が四元数型ではない（四元数成分取得）(pars:2285)
引数の数が 1 ではない（DES⇔IKE）(pars:2295)
引数が四元数ではない（DES⇔IKE）(pars:2296)
引数が不足（座標系変換） (pars:2305)
第一引数が整数型でない（座標系変換）(pars:2306)
引数の数が 2 を超過（座標系変換）(pars:2309)
第二引数が四元数型でない（座標系変換(pars:2310)
関数に引数がある（SQL 関数）(pars:2326)
型指定誤り（型名称）(pars:2396)
名称が不適切：予約語を用いた宣言等（変数、関数等の名称）(pars:2408)
多次元配列は宣言できない（[）(pars:2434)
関数の引数の型が不適（引数名）(pars:2463)
%c 変換で幅が 2 以上 （scanf）(pars:2646)
未定義の識別子（名称）(tbl:128)
識別子が重複している（名称） (tbl:168)
エスケープ文字の 16 進表現が不正（¥x）(tkn:362)
不正なトークン（トークン）(tkn412)
入力ファイルが開いていない nextCh() (tkn:440)

(err_s 関数への参照)
浮動小数二項演算結果の数値型が不明 (code:160)
四元数二項演算結果の数値型が不明 (code:181)
不正な左辺値 (code:192)

ゼロ除算(code:275)
main 関数がない (code:313)
関数の；または｛がない (pars:306)
関数の引数の型が不正 (pars:342)
本処理系では非 void 型関数の末尾には単独の return 文が必要 (pars:356)
main 関数が int 型でない (pars:368)
不正な break (pars:437)
case 式が定数式でない (pars:452)
case 式の値が重複している (pars:457)
対応する switch 文がない (pars:468)
default が重複している (pars:470)
do 終端の while がない (pars:550)
return 文に戻り値がない (pars:567)
void 型関数に値を返す return 文がある (pars:570)
関数の型と戻り値の型が異なる (pars:571)
シンボルテーブルパンク (pars:633)
COMMENT 文の位置が不正 (pars:667)
意図しない終了。‘}’が不足か？(pars:674)
continue がループ文内にない (pars:705)
同じ型の間で型変換しようとした (pars:738)
未実装の型変換 (pars:745)
二項演算の型調整不能 (pars:870)
&の次がアイデンティファイアでない (pars:897)
未登録のアイデンティファイア (pars:905)
＆の後の配列の数値型が不正 (pars:919)
添字指定がない (pars:924, 1105,1493)
演算子[+-*/]=の数値型が未対応 (pars:1125)
演算子[+-*/]=の前後の数値型が違う (pars:1131)
演算子[+-*/]=の種類が不正 (pars:1141,1152,1163)
不明な型 (pars:1169)
添字指定がない (pars:1293,1667,2420)
printf, logf の第一引数に書式文字列がない (pars:1430)
書式文字列の代入に対応する引数が無い (pars:1441)
printf の引数は２個まで(pars:1502,1549)
printf の第１引数が不正 (pars:1573)
scanf の引数は２個まで(pars:1672)

scanf の整数系書式が不正 (pars:1677)
scanf の浮動小数系書式が不正 (pars:1681)
scanf の QUAT 系書式が不正 (pars:1685)
TEXTURE(*[変数名でないもの]) (pars:1709)
TEXTURE(*[未定義の変数]) (pars:1717)
TEXTURE の引数が不正(pars:1720)
TEXTURE の引数の数が 1 ではない(pars:1728)
MATERIAL(*[変数名でないもの]) (pars:1744)
MATERIAL(*[未定義の変数]) (pars:1753)
MATERIAL の引数が不正(pars:1756)
MATERIAL の引数の数が 1 ではない(pars:1764)
想定外の EOF (pars:1960)
不正な添字 (pars:2427)
配列幅指定が整数定数式ではない (pars:2430)
不明な型 (pars:2448)
書式文字列：QUAT (pars:2560)
変数型誤り(void) (tbl:31)
記号表 over (tbl:37)
定数の文字数が過大 (tkn:306)
不正な文字定数(tkn:312)
文字列リテラルが長すぎる (tkn:392)
文字列リテラルが閉じていない (tkn:393)
不正な文字が使われている (tkn:405)
/*に対応する*/がない (tkn:468)
/*がないのに*/を検出した(tkn:474)
（文字列）の前に（文字）が必要 (tkn:526)
//トークン１が現れる前に、トークン２があるべきところ
(err_fi 関数への参照)
#コードが%d ステップを超えた (code:81)
メモリ不足：残り (code:222)
整数型引数を異なる書式で出力しようとした (code:1453)
浮動小数型引数を異なる書式で出力しようとした(code:1459)
四元数型引数を異なる書式で出力しようとした(code:1465)
printf の出力数値型が不適 (pars:1470)
(war_ss 関数への参照)
main()がない形式として再解釈 関数の外で変数が左辺値 (pars:230)

main()がない形式として再解釈 プログラムの最終行に到達 (pars:244)
右辺が NON_T conv (pars:734)
＊の後が整数型ではない（factor）(pars:947)
配列の添字が FLOAT 値 (pars:1091)
switch(kd)において kd が不正(pars:1194)
代入先がない scanf の書式文字列に変換%が指定されている(pars:1582)
FILE 関数はサポートしません cci_pars (pars:2006)
整数型の第%d 引数を浮動小数型に変換,_Q (pars:2265)

② 実行時のエラーメッセージ

コンパイルが成功し、データの読み込みを開始してから発生し、ログファイルに出力される実行時エラーを、表 2-4-2 に一覧する。

表 2-4-2 実行時エラー一覧

ゼロ除算(code:347)
ゼロ浮動除算(code:348)
プログラムカウンタが範囲外(code:415)
スタック・オーバフロー(code:426)
スタック・アンダーフロー(code:430)
time over(code:435)
スタックメモリオーバー：フレーム確保失敗(code:463)
配列の添字が負(code:618)
配列の添字が過大(code:619)
outfile が開いていない(code:705,736)
引数の無い PRINTF で、書式文字列に出力項目がある(code:718)
SCANF の QUAT 書式が不正(code:822)
PRINTF の QUAT 書式が不正(code:829)
printf(“%s”,adr)で、メモリ範囲外の番地が要求された(code:836)
GN テーブルパンク(dml:156,192)
オーバーフロー,fCOORD(dml:256)
オーバーフロー,COORD (dml:380)
オーバーフロー,TEXTURE (dml:425)
テクスチャ画像ファイルの入力に失敗 (dml:433)
オーバーフロー,TCOORD (dml:493)
オーバーフロー,NORMAL (dml:525)
オーバーフロー,COLOR (dml:537,565)
GetG<0 (code:617)
NG<=GetG (code:621)

GetG 失敗(code:626)
group_face 処理する面のテクスチャ ID が登録範囲外(dml:718)
group_face 処理する面の頂点 ID が登録範囲外(dml:739)
オーバーフロー,LINK (code:827,843)
LINK で親グループが見つからない(dml:850)
LINK で子グループが見つからない(dml:856)
LINK_XFORM の数値の数が不整合(dml:890)
LINK_XFORM の引数の数が指定方法と不整合 (dml:893)
LINK_XFORM の第二引数（順序）が不正 (dml:900)
LINK_XFORM の第三引数（パラメータ型）が不正 (dml:915)
TRANSLATE 引数の個数が不整合(dml:926)
ROTATE_X 引数の個数が不整合(dml:933)
ROTATE_Y 引数の個数が不整合(dml:938)
ROTATE_Z 引数の個数が不整合(dml:943)
ROTATE_A 引数の個数が不整合(dml:948)
SCALE 引数の個数が不整合(dml:956)
MATRIX 引数の個数が不整合(dml:963)
LINK の type が不正(dml:968)
LINK の order が不正(dml:974)
F()関数使用前準備不足(dml:1190)
Ft()で頂点が 3 未満(dml:1260)
Ft()準備不足(dml:1282)
Nx()関数使用前準備不足 (dml:1526)
Ny()関数使用前準備不足 (dml:1536)
Nz()関数使用前準備不足 (dml:1546)
Cr()関数使用前準備不足 (dml:1672)
Cg()関数使用前準備不足 (dml:1682)
Cb()関数使用前準備不足 (dml:1692)
Ca()関数使用前準備不足 (dml:1702)
範囲外(Lm) (dml:1767)
リンクが未選択(Li) (dml:1779)
リンクが未選択(Ls) (dml:1801)
リンクが未選択(Lr) (dml:1817)
リンクが未選択(Lt) (dml:1838)
リンクの子グループの ID が負(Lc)(dml:1918, sql:1005))
リンクの子グループの ID が過大(Lc)(dml:1919,sql:1006)

VERTEX に５番目以降の引数（face:91）
s (face(97,102)
頂点リストを格納するメモリ取得失敗(face:126)
FACE コマンドでスタック・アンダーフロー(face:184)
FACE,LINE のリスト FL 伸長失敗(face:224)
FACE_VERTEX の引数が不足（face:250)
FACE_VERTEX 失敗（face:275)
GROUP_FACE でスタック・アンダーフロー（face:318)
GROUP_FACE 処理で、参照するグループ ID が過大（face:328)
GROUP_FACE 処理で、参照するグループ ID が過小（face:334)
GROUP_FACE 処理で、参照する面の ID が過小（face:344)
GROUP_FACE 処理で、参照する面の ID が過大（face:350)
GROUP_FACE 処理で、参照する面の頂点 ID が不正（face:368)
GROUP_FACE 処理で、参照する面の頂点 ID が未定義（face:377)
GROUP_FACE 処理で、頂点リストが空（face:398)
未定義の面[ID]を参照(FACE_COLOR), NF=面総数(face:424)
未定義の色[ID]を参照(FACE_COLOR),NC=色総数(face:427)
未定義の面を参照(FACE_NORMAL) (face(454)
未定義の法線を参照(FACE_NORMAL) (face:455)
未定義の面を参照(FACE_TEXTURE)(face(468)
未定義のテクスチャを参照(FACE_TEXTURE) (face:469)
未定義の面を参照(FACE_MATERIAL) (face:486)
FP==NULL,GETC() (file:71)
FP==NULL,WATC() (file:80)
FP==NULL,INTEGER() (file:92)
FP==NULL,FLOATER() (file:108)
FP==NULL,SIORI() (file:122)
FP==NULL,SEEK() (file:130)
書式文字列への参照アドレスが NULL(inputdata) (file:222)
書式文字列の参照が「%」ではない (file:226)
FP==NULL,SCANF() (file:400)
scanf で、取得した整数値の格納先がない（file:415,425)
scanf で、取得した浮動小数値の格納先がない（file:430)
scanf で、文字列型の格納先がない（file:435)
シンボルテーブルパンク(file:458)
scanf で、数値を取得しないのに格納先がある（file473)

SCANF の QUAT 書式が不正(file:540)
scanf で、取得した四元数値の格納先がない(file:551)
オーバーフロー,COORD (ip:176)
LINK_XFORM のタイプ指定が不正(ip:501)
#LINK_XFORM の引数とタイプが不整合(ip:504,sql:1411)
LINK_XFORM の第二引数（順序）が不正(ip:510,sql:1417)
LINK_XFORM の第三引数（パラメータ型）が不正(ip:522,sql:1430)
//'Strdup fail (misc:207)
Free(NULL) (misc:311)
LINK_XFORM の数値の数が不整合(sql:1408)
TRANSLATE 引数の個数が不整合(sql:1437)
ROTATE_X 引数の個数が不整合(sql:1444)
ROTATE_Y 引数の個数が不整合(sql:1449)
ROTATE_Z 引数の個数が不整合(sql:1454)
ROTATE_A 引数の個数が不整合(sql:1459)
SCALE 引数の個数が不整合(sql:1467)
MATRIX 引数の個数が不整合(sql:1474)
LINK の type が不正(sql:1479)
LINK の order が不正(sql:1488)

③ メタファイル中の logf 関数によるメッセージ出力

メタファイルのプログラマは、logf 関数を用いて、実行中の処理状態を確認するメッセージをログファイルに出力することができる。デバッグが終了し最終的に完成したメタファイルからは削除される。logf 関数の引数は、書式文字列と出力データから成り、printf 関数とほぼ同等である。利活用形態として、VC-2V、VC-4D においてメモリ上あるいはデータベース上の町並み、建物等を読み出して、ファイル出力を行う場合には、printf 関数が最終的にファイル出力されるデータファイルへの出力を行うのに対して、logf 関数は、デバッグのためのログファイルへの出力を行う。

（５）配列の長さ等の最適化

本処理系におけるメタファイルは、あるファイル形式で記述された様々な不特定のデータファイルを処理するための汎用のものである必要はなく、添付対象とする特定のデータファイルを変換できれば十分であり、むしろ無用なメモリの浪費は少ない方が望ましい。

このため、処理中に生成する配列等の必要幅は既知であり、メタファイルの冒頭で宣言する定数値も最小限とすることが望ましい。

よって変換処理が正しく行われたことを確認した上で、処理終了時点でのリストの長さ等を把握し、これに基づいて冒頭の宣言において配列幅を設定する定数に反映させ修正することにより、過大なメモリを配列の領域確保のために浪費することを防ぐことができる。

（６）不要な関数等の削除

仕様書・定義書に定められているデータ形式を用いた保存ファイルであっても、通常はその中で使用されているキーワードやコマンドは、用意されている利用可能なものの一部にすぎない。従って、変換処理を行った結果、参照されることが一度もなかったようなキーワードやコマンドに対応するメタファイルの処理（関数等）は、メタファイルから削除することにより、メタファイルを小さく簡潔にすることができる。

逆に、新たにメタファイルを作成するデータ形式に関しては、最初にまず参照されているコマンドの一覧を作成し、それらについて処理内容を実装する方法が効率的である。データファイルを作成する際に使用したアプリケーション（CAD、モデラー等）が自動的に出力するコマンドで、モデリング作業に無関係にデフォルト値を宣言しているだけのような、意味の無いコマンドに関しては、そのコマンドに関連した引数パート等を最後まで読み飛ばすだけの処理を作成すれば良い。

（７）データファイルの真贋判定、改竄検知

メタファイルは、添付するデータファイルが特定のものであることから、真贋性に関する検査の処理を付け加えることができる。ファイルの長さ、チェックサム、あるいはより高度なデータファイル固有の値をデータファイルに関して解析し、その値がオリジナルのデータファイルと一致することを確認することにより、メタファイルとデータファイルの対応関係を確認することができる。この固有の値は、定数値としてメタファイルの中に書き込むことができる。ごく簡単なファイルサイズの検査、チェックサムの処理例をリスト 2-4-1 に示す。より高度な検査をメタファイルで記述することも可能である。

但しこの方法が有効であるためには、データファイルとメタファイルの双方が同時に偽造されていないことを保証する必要があり、そのためにはメタファイルをデータファイルと関連づけられた別の場所に非公開で保管するような管理が必要であろう。一般にメタファイルは、データファイルよりも遥かにサイズが小さい。

リスト 2-4-1 データファイルの偽造・改竄の検出処理

```
int check(){
  int c,sum;
  if( LEN() != 固有の値) return 0; //長さの検査
  for(sum=0; 0 <= (c=GETC()); sum += c); //チェックサム検査
  if( sum != 固有の値) return 0;
  ・・・その他の高度な検査・・・
  return 1;
}

int main(){
  ・・・・
```

```
    if( !check() ) logf("データファイルは改竄されている");
    ・・・・
}
```

２－５．三次元アーカイブスの作成

（１）携帯端末用ロードデータの作成

① 選択用リスト modelindex.txt の作成

保存データと、これを解読するための処理を定義したメタファイルの組を、必要な数だけ modelindex.txt ファイルに登録することにより、VC-3M（むかしめがね）を用いて現場で表示することが可能となる。その際に、モデルを記述する原点の緯度・経度・標高を併せて指定することにより、記録された建物等を画面上で過去に存在していた位置あるいは建設が計画されていた位置に表示することができる。この一覧データは、csv 形式で記録されており、１つの行は、

表示名,メタファイル名,データファイル名,緯度,経度,標高

を含んでいる。表示名は、モデル選択画面（「見たい場所」）で一覧表示される、各場所の名称である。緯度、経度、標高は、モデルを記述している座標系の原点の位置を示している。

奥尻島の３地区を登録したデータの例を、表 2-5-1 に示す。

表 2-5-1 奥尻島の表示データのための、modelindex.txt の例

みさきこうえん,LSSG.cmm,misaki.geo,42.0524,139.4505,2.0
みさきこうえん津波２,LSSG.cmm,misakitsunami2.geo,42.0524,139.4505,2.0
はまかぜこうえん０,LSSG.cmm,hamakaze2.geo,42.060198,139.4494,0.0
はまかぜこうえん６津波,LSSG.cmm,hamakaze6t.geo,42.060198,139.4494,0.0
よねおかけんえい,LSSG.cmm,yoneoka.geo,42.0682,139.44435,0.0

2014 年 10 月 8 日に現地体験教室に使用した際のファイルであり、図 2-1-3 に示した一覧表示画面に対応している。この緯度、経度は、あらかじめ図上で計測した上で、現地において地図上で確認できる場所に赴き、VC-3M をセットアップした携帯端末を用いて計測を行って確認した。

古写真から復原したデータの場合には、テクスチャファイルを texture ディレクトリの下に用意しておく必要がある。

なお、建物等を記述するデータの作成に使用した座標軸の原点位置の緯度・経度・標高は、この一覧ファイルに記述する以外に、モデルデータの中に記録されている数値を利用することも、メタファイルの中で記録することもできる。また、シャッターを操作した際に記録され、mobile.ja.scn ファイルとして保存されるデータの中にも、撮影場所の緯度、経度、標高が記録される。詳細は、３－５で解説する。

② データファイルおよびメタファイルの格納

携帯端末の内蔵 SD カードの下に、VirtualConverter という名称のディレクトリを作成し、この直下に上記の modelindex.txt を格納する。このディレクトリには、シャッターボタンが押された場合の時刻、場所、姿勢（カメラアングル）を記録した mobile.ja.scn ファイルが作成される。また、データをロードする際のエラーメッセージ等を記録したログフ

ファイル log.txt が作成される。

便宜のためアプリをセットアップするための VC-3M.apk ファイルもここに置いている。

このディレクトリの配下に、サブディレクトリとして以下のものを作成し、それぞれに必要なデータを格納する。

1) data：データファイルを格納する。一つの建物が一つのファイルである。それぞれの建物を記述するデータファイルは、異なる形式であってもかまわない。上記の modelindex.txt に登録されているメタファイルとデータファイルが対応していれば読み込むことができる。
2) meta：メタファイルを格納する。データファイルの形式が同一であれば、一つのメタファイルで複数のデータファイルに共通に使用することができる。
3) texture：データファイルの表示にテクスチャを使用する場合には、画像ファイルを格納する。
4) sensorvalue：デバッグのために、センサーの計測値を記録したファイルを格納するためのディレクトリである。初期状態で空であって構わない。
5) image：シャッターボタンが押された場合に、背景画像を JPEG 形式のファイルとして保存する。初期状態で空であっても構わない。
6) thumbnail：保存した背景画像を縮小したサムネイル画像を保存する。記録再生を行おうとする場合に、マトリクス表示する背景画像として使用する。ディレクトリが存在していれば、初期状態で空であっても構わない。

（２）アーカイブスを使用した現場活動の記録の保存

現場での閲覧の後に、以下のファイルが作成される。

① mobile.ja.scn

背景画像名、シャッター時点での GPS 座標、端末の姿勢、キャリブレーション値を記録する。VirtualConverter ディレクトリ直下に作成される。

初期状態で存在しなければ、最初の終了時点で新たに作成される。

既存のものがあれば、シャッター操作により、末尾にデータが追記され、アプリ終了時点で保存される。終了前に異常終了（電池切れ等）した場合には、記録は失われる。また、記録再生画面で削除操作が行われた場合には、対応する記録は削除される。但し、関連する背景画像ファイル等は温存される。

② Images ディレクトリの下に作成される背景画像

シャッターボタンが押された時点で記録される背景画像(jpg 形式)である。

ファイル名称は、撮影日時・時刻に基づき、以下のように自動的に決定される。

[例]　BI20141009.095919.jpg

[解釈]　2014 年 10 月 9 日 9 時 59 分 19 秒に保存された背景画像(Back Image)

③ thumbnail ディレクトリの下に格納される見出し用縮小画像

TBI20141009.095919.jpg

2014 年 10 月 9 日 9 時 59 分 19 秒に保存された画像の縮小ファイル(40-50KB 程度)

④ log.txt

仮想コンバータが起動する度に、言い換えると「見たい場所」が選択される度に作成される。メタファイルのコンパイル・エラーや、実行形式のランタイム・エラーを記録する。

utf-8 形式で作成されており、エラーが生じた場合には、このログファイルを表示して終了する。

エラーが存在しない場合でも、作成されるが表示はせず、そのまま表示処理に進む。動作状況を把握するために使用することができる。

（３）WEB 配信用の圧縮ファイルの作成

公開可能なアーカイブスに関して、携帯端末に実装するデータを配信するためには、必要とするファイルを一つのファイルに圧縮・梱包したファイルを作成するのが便利である。圧縮・解凍の作業を行うためには、携帯端末側で利用するファイル・マネージャ等に、ディレクトリ単位の圧縮と解凍を行う機能が備わっており、ZIP 圧縮が用いられている。

携帯端末で VC-3M と三次元アーカイブスをロードし実行するためには、VirtualConverter ディレクトリの内部に、以下のファイルが用意されている必要がある。

① ModelIndex.txt（見たい場所、建物などの一覧）

VirtualConverter ディレクトリ直下

② Meta ディレクトリの下に格納されたメタファイル

データファイルのそれぞれを解読するための処理を定義したメタファイルである。

一つのメタファイルを、同一形式の複数のデータファイルに共通に用いることができる。

また、固定形状のデータをメタファイルだけで記述することもできる。

③ Data ディレクトリの下に格納されたデータファイル

様々な記録形式で建物等の三次元形状を定義したファイルである。

④ Texture ディレクトリの下に格納されたテクスチャファイル

建物の屋根面、壁面等にテクスチャ画像を定義する場合には、このディレクトリに格納する。

⑤ VC-3M.apk

アプリのセットアップファイルで、VirtualConverter ディレクトリ直下に置かれる。但し、このファイルがどこにあってもセットアップは可能であり、既にセットアップが行われている場合には、新たなデータのロードに際して、再セットアップは不要である。

（４）長期保存媒体への記録と再生

上記のように、建物や町並を記録した三次元データの解読方法がメタファイルにより記述され、さらにローカル座標系の原点位置の緯度・経度・標高が明らかであれば、将来時点において、同じ場所にデータ表示することが可能であり、表示以外にも様々な利活用方法が想定できる。そのためには、長期保存が可能な記録媒体への書き込み・読み出しが必要である。

本研究においては、２－６、２－７において、RAID１等によりメンテナンスが保証されているサーバー上に保存する方法を解説すると共に、頻繁なアクセスが期待されない、いわゆるコールドデータに関しては、ガラス等の長期保存可能な媒体に記録する方法をテストした。具体的には、Housemap.dll という、住宅に関する画像データ等を扱う、景観シミュレータのためのプラグインを作成し、その機能の一部として、データファイルとメタファイルの組を、二次元のビットパターンに変換して、レーザー加工機を駆動するためのデータとして出力する処理を行った。この記録を含む記憶媒体（電子棟札、電子定礎等とも呼ぶべきもの）を将来読み出す時点では、イメージスキャナでビットパターンを取り出す。これについては更に実用化に向けて研究・試作中であり、本書では割愛する。

２－６．サーバとデータベースの管理

（１）はじめに

仮想コンバータ VC-4D の開発を行った 2012 年当初、OS として Windows2003Server（32bit）を搭載した WEB サーバ上でのコーディングと動作テストを実施し、任意形式の三次元データファイルと、データ記録形式を記述したメタファイルの組をアップロードし、SQL サーバ上に座標値、図形、属性等を展開するアップロード動作を確認した。更に、ユーザがリクエストに添付したメタファイルに従い、別の形式の三次元データとして再構築して配信する動作を確認した。

Microsoft 社による Windows2003Server のサポートが 2015 年 7 月 15 日に終了したため、国総研が景観シミュレーションシステムのダウンロードサービス(1996-)や、まちづくり・コミュニケーション・システム(2001-)およびこの機能を活用した三次元アーカイブスの公開、VC-4D の開発(2012-)を行っていた電算室の公開サーバの OS を Windows2012Server（x64）に更新すると共に、SQL データベースも MSSQL7 から SQL2012 に更新した。この移植に際して、OS のセットアップを仮想ハードディスク（VHD）上とし、可搬性を高めた。この移植そのものは一般的な手順書に解説されているものであって特殊性はないが、古い処理系とデータを引き継ぎながら運用しているサーバ機能の新たな OS への移植に際して遭遇した問題点と解決手段は、今後の新たなサーバへの開発成果の導入や、既存機能とサービスを維持しながらの OS の更新に際して有益な経験となると考え、以下に記録しておく。

（２）仮想化

①概念と用語

Virtual は通常「仮想」と訳される。「空想」や「虚構」という意味はなく、異なるハードウェアや OS 環境下でも、同じようにソフトウェアが動作する可搬性を実現することを意味し、実体はどこかに必ず存在する。以下のような用語がよく用いられる。

１）パーティション

物理的なハードディスクの上に複数の領域を確保し、それぞれが別のドライブとして OS からアクセスできるようにする方法である。ハードディスクの先頭 512 バイトにある MBR(Master Boot Record)にパーティションの定義が記録されている。MBR で定義できるパーティションは４までである。

なお、MBR による方法では最大２TB までしか管理できないが、これに代わる GPT(GUID (Global Unique IDentifier)Partition Table)を導入したハードディスク装置では 128 のパーティションを作成できる。但し、Windows Vista 以後の 64 ビット OS が適合する。

a. システム・パーティション

保護のためエクスプローラでは表示されず、ドライブ名も無い。

ハード関連のブート処理に使用する。

コンピュータに一つだけあり、起動する複数の OS の選択リスト等が置かれる。

b. ブート・パーティション

セットアップした仮想マシン（OS）の数だけ作成される。

各 OS の起動に必要なシステム・ファイル群を格納する。

c. アクティブ・パーティション

選択され起動された OS のブート・パーティションである。

「ディスクの管理」（GUI）では、パーティションをアクティブとしてマークする。

「diskpart」（コマンドライン）では、セレクトした上で、アクティブにする。

```
>select volume 0
>active
```

d. プライマリ・パーティションと拡張パーティション

新しいパーティションを作成する際に選択する。

プライマリ・パーティションは、システム起動が可能なパーティションである。

作成すると、一つのドライブ名が割り当てられる。

拡張パーティションは、システム起動ができないパーティションである。

あるパーティションの領域を「拡張パーティション」として位置づけた上で、その内部に複数の論理ドライブを作成することができる。拡張パーティションを作成しただけではドライブとして認識されず、その内部に論理ドライブを作成すると、ドライブ名が割り当てられる。数の制限はないが、名前に使用できるアルファベット文字の数で制約される。

２）BIOS　Basic Input/Output System の略

1980 年代の PC においては、ディスクドライブの読み書きなどの基本処理をライブラリとして EPROM に格納したものを BIOS と呼び、プログラムから呼び出すと共に、起動時の OS の読込み（IPL）もこの中に記述していた。その後 OS が拡大し基本処理の大半が ROM に置かれなくなってからも BIOS の名称が存続しているが、実質的な内容はセルフテストや起動条件設定などを含むファームウェアとして、書き換え可能なフラッシュメモリに記憶しているハードが多い。更にこれに代わる UEFI(Unified Extensible Firmware Interface)を搭載したボードも製品化され、3TB のドライブからの起動を可能としている。

OS 自体のロードは、FDD、HDD（その各パーティション）、CD/DVD-ROM、Ethernet、更にはフラッシュメモリなどの USB デバイスからも選択的に行うことができ、起動を可とするデバイスと優先順位を BIOS の設定画面で変更することができる。

３）Boot Menu 起動デバイスの選択

OS をロードするためのプログラムをロードする方法を、靴のつまみに喩えて bootstrap（略して boot）と呼ばれる。通常の起動では BIOS で設定した優先順位に従い自動的にロードされるが、前回異常終了した場合や、起動デバイスとして選択されているデバイスが壊れている場合などに、オペレータに選択を求めるメニューがデバイスの一覧として表示される。メンテナンスのため起動時に指示表示されるファンクション・キー等を操作する

ことにより選択可能なドライブの一覧が表示され選択可能となる。例を示すと、

1. CD-ROM Drive

2. Removable Devices

3 +Hard Drive（内蔵 HDD からの起動）

4. MBA V7.7.5 Slot 0200　（MBA: Multiboot Agent を用いたネットワークからの起動）

４）Network Boot

BIOS による起動方法選択の一つで、内蔵ドライブではなくネットワークを介してマシンを起動する。ハードディスクが高価であった 1980 年代の端末起動技術を、セキュリティの高い端末起動やリモート操作でのＯＳ導入に活用したもの。

５）PXE　：(Preboot Execution Environment)

ネットワークからの起動のための規格。

Broadcom UNDI PXE-2.1 v7.7.5 を選択した場合、

CLIENT MAC ADDR:xxxxxx　GUID:yyyyyy

DHCP　と表示して、OS 取得先のホストを探しに行く。

(Broadcom は会社名, UNDI は Universal Network Device Interface)

６）Multi Boot　マルチブート

マシンにセットアップされている複数の OS から一つを選んで起動する。Windows NT では NTLDR、Vista では Windows Boot Manager が標準装備されていた。それぞれの OS は一つのパーティションを占有するため、OS 毎に用意する必要があった。

７）VHD（Virtual Hard Disk、仮想ハードディスク、②で解説）

Windows7 以降は、一つのバイナリ・ファイル（拡張子.vhd）としてハードディスクを仮想的に表現することにより、一つの物理的ドライブ（パーティション）の中に、容量が許す限りいくつでも新たな仮想ハードディスクを追加することができ、より多くの OS を選択的に起動することができるようになった。最大 2 TB。

ある OS 上で、一つの VHD ファイルをマウント（アタッチ）すると、あたかも一つのドライブを接続したように内部のファイルにアクセスすることができる。

８）Virtual PC

一つの 64 ビット OS の上のアプリケーションとして別の 32 ビット OS を起動し同時に実行させる。Windows7、2008R2 から導入され、Hyper-V への移行に伴い廃止された。

９）Hyper-V は、64 ビットのマルチブート環境を実現する。

Windows8、2012 から、VirtualPC に代わり導入されたハードウェアの上の階層（仮想環境）で複数の VHD 毎に OS をセットアップし、同時に起動し実行することができる。VHD は最大 64TB に拡張された（拡張子.vhdx）。ホストとなる管理 OS 自身も、この仮想環境の上に載っている点が VirtualPC とは異なる。

一つの OS はマウント（アタッチ）された複数の VHD にアクセスでき、複数の OS 間で VHD を共有することもできる。一方 Hyper-V でマシンを仮想化した場合、ディスク以外

の古いハード（シリアルポート、マウスポート、IDE 接続の FDD、CD/DVD 等）はエミュレーション化の影響で効率が下がる点がマルチブートの起動の場合とは異なる。

１０）Power Shell

MS-DOS の command.com→Windows NT の cmd.exe と類似した外観のコンソール画面を提供する実行形式(PowerShell.exe)。仮想マシンをブートメニューに追加する処理などを実行できる。

１１）Sysprep （sysprep.exe）

OS からマシン固有の ID である SID 等を除去して一般化した VHD を作成する。別マシンの複製に際し新たに固有の値を与え、ドメイン参加に際しての重複障害を回避する。

②仮想ハードディスク

ハードディスクの上に作成された一つのパーティション（ドライブ）は、通常はディスク管理ツールにより作成する。Windows では一つのパーティションに複数の OS を並存させることはできないため、新たなブートパーティション(または物理的なドライブ)を増設しなければ、一台のサーバ上に複数の OS を併存させ、移植前の OS と移植後の OS によるマルチブートの環境を実現することはできない。

これに対して、Windows7, Server2008 以降に利用可能となった仮想ハードディスク(VHD: Virtual Hard Disk)では、既存のドライブ上に大きなバイナリ・ファイルを作成し、その内部を読み書きすることにより、恰も独立した増設ドライブのように扱うことができる。このような仮想ハードディスクへの OS のセットアップ方法は、Windows7、2008Server 以降で可能となった。具体的には、インストール DVD-ROM をドライブに装着して一連の作業を開始し、必要なファイルが一時的なディレクトリに解凍された後、セットアップ実行形式を開始する前のタイミングで選択を行い、DVD が提供するコマンドライン・コンソールを用いて、DISKPART ユーティリティを起動し、これを用いて新たな VHD を作成し、セットアップ中の仮設的な OS にこれをアタッチして、そこに新たな OS を導入する。

仮想ハードディスクは、例えば VC-3M の開発・デバッグに用いた Windows マシン上での Android 携帯端末のエミュレータの内部の記憶装置(実際の携帯端末では SD カードに相当する、４－５参照）としても使用されており、また、Windows におけるハードディスクのバックアップもまた同様の形式のバイナリ・ファイルとして実現されている。

一つの仮想ハードディスクのファイルをデタッチして別のマシン（ディスク）にファイルとしてコピーし、そこで再びアタッチすることにより、物理的なドライブや記憶媒体をあるマシンから取り外して、別のマシンに装着するような結果を、コンソールで行う作業により得ることができる。

③実施例

具体的には、Windows7、Windows2008R2 以降の OS において、コンソールから diskpart コマンドで作業を行うか、ディスク管理ツールを用いて作業を行う。固定長と可変長のドライブを作成することができる。なお、Windows7 以降では、既存のパーティションのファ

イルを維持したまま、物理的なパーティション（ドライブ）のサイズを変更することもできるようになった。

Windows7 のセットアップディスクからブートし､ディスクの修復メニューでコマンドラインを選択すると､X ドライブ上に導入されたコマンド群が利用できるようになる。ここから diskpart コマンドで VHD を作成し、そこに OS をセットアップすることができる。

マルチブートを追加するためには、

a. diskpart ユーティリティにより VHD を作成し、そこに OS をセットアップする

b. bcdedit によりブートリストに追加する(BCD: Boot Configuration Data)

Windows7 マシンに、既存の OS を残したまま VHD を作成し、Windows2012Server を VHD 上に導入することができた。その手順を以下に例示する。

1）DVD－R から起動できるように BIOS を設定して、Windows2012 サーバセットアップ用 DVD から起動する。

2）セットアップ開始前の画面で、ディスクの修復を選択し、コマンド画面を開く。

3）DISKPART を起動する

4）空きディスクに VHD を作成する（CREATE コマンド)。

5）作成したディスクを選択する。

6）VDISK をアタッチする。

7）コマンドラインまで戻り、DVD>setup

setup は、コマンドラインとして Windows 2012 Server の上で動作しており、アタッチした VDISK（VHD）にアクセスすることができ、警告は出るものの、ここを選択して OS をセットアップすることができる。以後の手順は、通常のセットアップと同様である。

重要な点は、setup.exe が実行されている OS において、セットアップ先の VHD が見えていることが必要である点であり、このために、setup の直前に、セットアップに用いている OS のコマンドラインでアタッチ操作を行った後、再起動することなく、その OS の上で引き続きセットアップを完了させる。

④効果など

VHD は、これを使用していない OS から見ると、その OS にアタッチしない限り単なる一つのバイナリ・ファイルであり、その中にある構成要素を直接操作することはできない。よって、セキュリティ上の分離は高まっている。

VHD をファイルとして別システムに移動し、そこでアタッチすることにより、容易に移築することができる。

セットアップにおいて、VHD を作成し、そこに新たな OS を導入する方法は、ハードディスクを増設したり、新たなパーティションを作成したりできない場合でも実行できる。Windows 7 、2008R2 以降の OS がセットアップできれば、OS 導入後に、これらの機能を用いて既存のパーティションの内容を保持したまま、サイズだけを変更することもできる。

更に、過去に固定的なパーティションに導入した OS に関しても、これを VHD の上にコ

ピーした上で、そこから起動するように再構成することができる。

このような操作により、過去のテキスト、画像、プログラムおよびそれらにより構築された WEB サイトをアーカイブした上で新たな OS 上に移植する際に、古い OS を含めた古い環境全体を再実行可能な古い動体として凍結保存することが容易になった。言い換えると、一つの環境全体が記憶媒体の上での一つのファイル VHD として記録保存される。

（３）FTP

①概念と用語

FTP とは、File Transfer Protocol で、本処理系においては、IIS(Internet Information Service, Ver.8)によりサービスが提供される。HTTP プロトコルと並んで、古くから使用されてきたネットワーク上のファイル転送方法である。便利な機能ではあるが、近年では利便性よりもむしろ、セキュリティ上の脆弱性が問題となっており、不正なアクセス等を防止することが課題となっている。

FTPS とは、SSH File Transfer Protocol で、FTP において暗号化されていない状態において送信されるユーザ名とパスワードが盗聴されることを防ぐ方法である。

Explicit モードでは、ハンドシェイク完了後に暗号化された通信を行う。

Implicit モードでは、ハンドシェイクから暗号化された通信を行う。

FTPS サービスは、サーバ側で IIS により提供することができるが、クライアント側で Windows コマンドの FTP ではサービスを受けられない。FFFTP などにより初めて可能。UNIX 系の lftp をクライアントとして使用することは可能。また、FTPS サービスを実行するためには、サーバ側で SSL 証明書（有償）を購入する必要がある。ただし、SSL サイトと共通の証明書をワイルドカード証明書として利用することは可能。

通常は、クライアント側から PORT コマンドに引数としてクライアントのＩＰアドレスとポート番号を指定して送付し、データ転送を開始する。パッシブ接続では、クライアント側から PASV コマンドを送付して返されたサーバのＩＰアドレスとポート番号を用いてデータ転送を開始する。NAT や IP マスカレード処理に際して、PASV コマンドに対するレスポンスは変換対象とならないため、バグになりにくい。

NAT(Network Address Translation)は、ルータやゲートウェイが IP アドレスを変換する処理である。ルータは異なるネットワークの間を中継する。ゲートウェイはプロトコル変換を行う。たとえば無線 LAN では IPX/SPX プロトコルと TCP/IP プロトコルの間の変換を行う。

②ＩＩＳの設定

http、ftp プロトコルで Web コンテンツなどのファイル配信を行う IIS(Internet Information Service)サービスの導入は、Windows2012Sever においては「サーバの管理」画面から「機能と役割の追加」画面を開いて行う。IIS と、その下の HTTP、FTP 等を選択した上で実行すると、セットアップ DVD を使用することなくサービスを開始することができる。

FTP サイトを開設した上で、以下の設定を行う。

1）IP アドレスとドメインの制限

機能設定で「許可」を指定すれば、特定のＩＰアドレス以外からのアクセスはすべて許可される。本処理系では「拒否」を指定した上で、隣接するマシン（デバッグ用）の IP アドレスと、ファイアーウォールの IP アドレスを許可する。ファイアーウォールが複数あって負荷分散している場合には、すべて記述しておかないと、大きなファイルの転送に支障がある。

2）SSL 接続

FTPS を使用しない場合は、「許可」とするだけでよい。「必要」を選択すると、暗号化しない FTP 接続ができない。

3）承認規則

接続するユーザを指定する。「すべてのユーザ」に対して、読み取りと書き込み許可を与えるか否かを指定する。特定のユーザに対して特別な許可を与えたり、禁止したりすることができる。

4）ファイアーウォールのサポート

FTPS を使用する場合、またはファイアーウォールでパケットフィルターを処理していない場合に、パッシブ接続を受け入れるには、ファイアーウォールの外部 IP アドレスを設定する。このアドレスは、外部から見たときのファイアーウォールのアドレスである。

5）ユーザの分離：特に行っていない

6）認証

基本認証：有効（規定のドメインは空白のまま）

匿名認証：無効

（有効にする場合、アカウントとパスワードを設定する）

IIS への設定は、FTP サービスを再起動した時点で有効となる。

③アカウントの設定

FTP 接続する際に、クライアント側で入力する管理用のユーザ名とパスワードを決め、これをサーバのユーザとして登録する。FTP 接続専用に、管理権限の弱いアカウントとパスワードを一つ作成し、ファイル転送を行うディレクトリのファイル送受信（つまり読み取り・書き込み・変更）に関してのみ強い権限を与えるのが便利で安全である。

匿名接続を許可すると、任意のユーザ名＋パスワードで進入できるため、安全ではない。基本接続を使用する場合には、あらかじめ用意したアカウントとパスワードでのみアクセスが可能となる。但し、接続時の認証過程で、交信されるアカウントとパスワードが暗号化されないために、回線の状況によっては盗聴される危険性が存在する。このためには SSL 接続が勧奨されている。SSL 接続を行うためには、サーバの認証をプロバイダから有償で取得する必要があるが、自己認証を用いて無償で実現する方法は存在する。

④ディレクトリ・セキュリティの設定

FTP 接続する際に、クライアント側で入力するユーザ名の、サーバ上の各ディレクトリに対するアクセス権を設定する。FTP 接続でログインするユーザ名に対して、読み取り・書き込み・変更を許す権限を設定する。システム構築／再構築段階では、様々な要因により通信が成立しない状況が想定されるため、問題を単純化するためにまず、Everyone-Full Control 状態で動作を確認した後に、Everyone アカウントを削除して FTP 専用のアカウントだけを進入させるように変更する。

⑤ファイアーウォールの設定

Windows2012Server の場合、「セキュリティが強化されたファイアーウォール」ツールを使用する。起動方法は、[コントロールパネル]→[セキュリティ]→[ウィンドウファイアウォール]→[詳細設定]という経路、または[管理ツール]→[セキュリティが強化されたファイアーウォール]という経路を辿る。

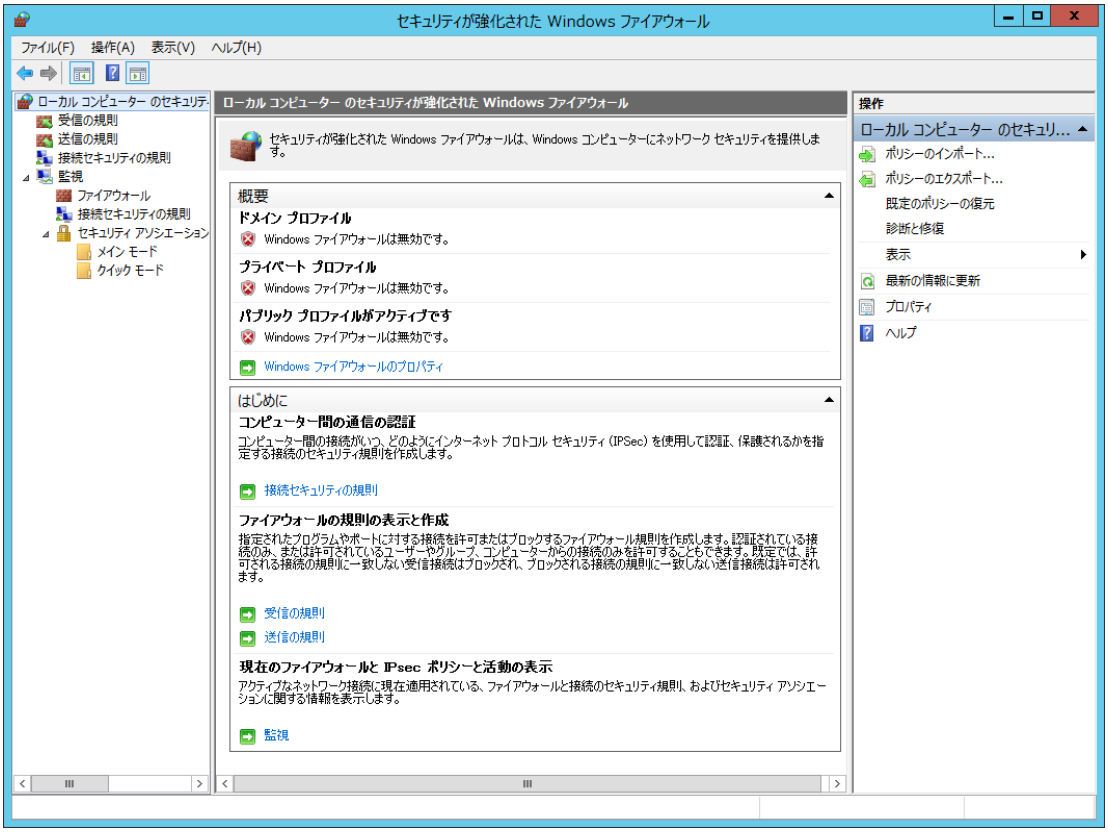


図 2-6-1 ローカルコンピュータのセキュリティ設定画面（Windows2012Server）

ファイアーウォールの有効・無効の切り替えは、右側のプロパティで行う。

FTP の受信に関する規則では、左ペインで「受信の規則」を選択した上で、中央ペインで FTP に関連する項目をダブルクリックまたは右クリック→プロパティまたは左クリックで選択状態にした上で、操作メニューのプルダウンからプロパティ選択し、編集画面をポ

ップアップする。

FTP 受信接続に関しては、[1]FTP サーバ（FTP トラフィック）、[2]FTP サーバセキュリティ（FTP SSL トラフィック）、[3]FTP サーバパッシブ（FTP パッシブトラフィック）を設定する。

FTP の送信に関する規則では、左ペインで「送信の規則」を選択したうえで、同様の手順で詳細設定画面を開く。

FTP 送信の規則に関しては、[1]FTP サーバ（FTP トラフィック送信）、[2]FTP サーバセキュリティ（FTP　SSL トラフィック）を設定する。

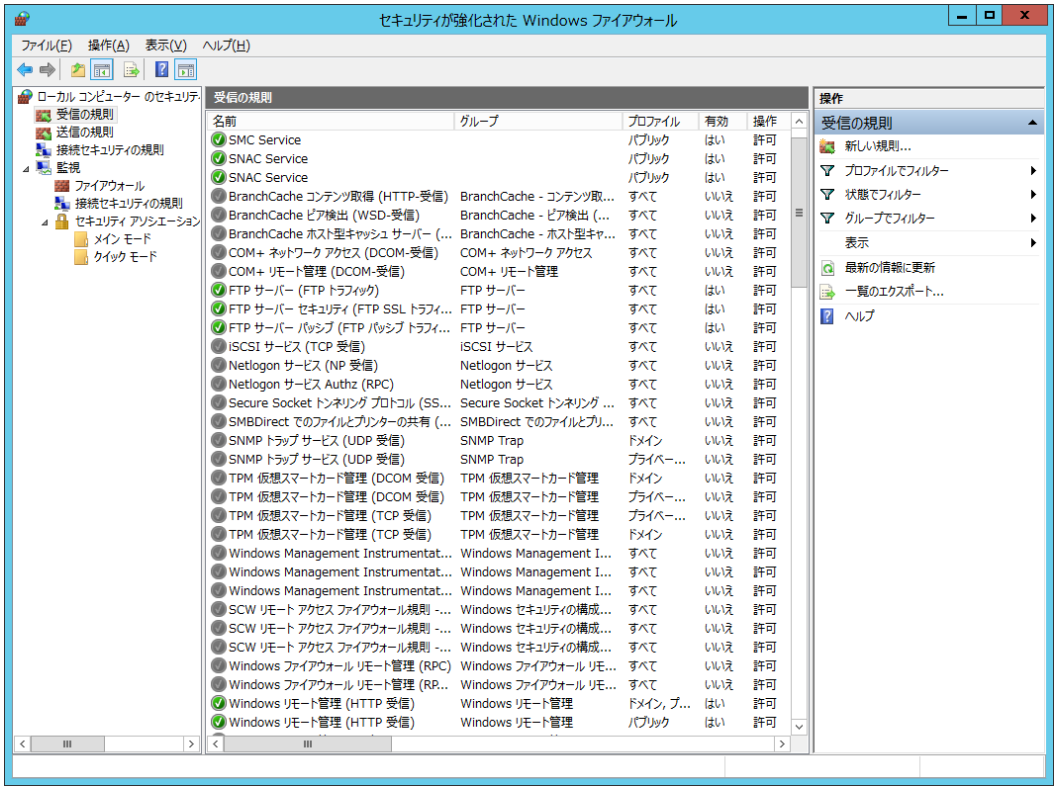


図 2-6-2 受信の規則の設定画面（Windows2012Server）

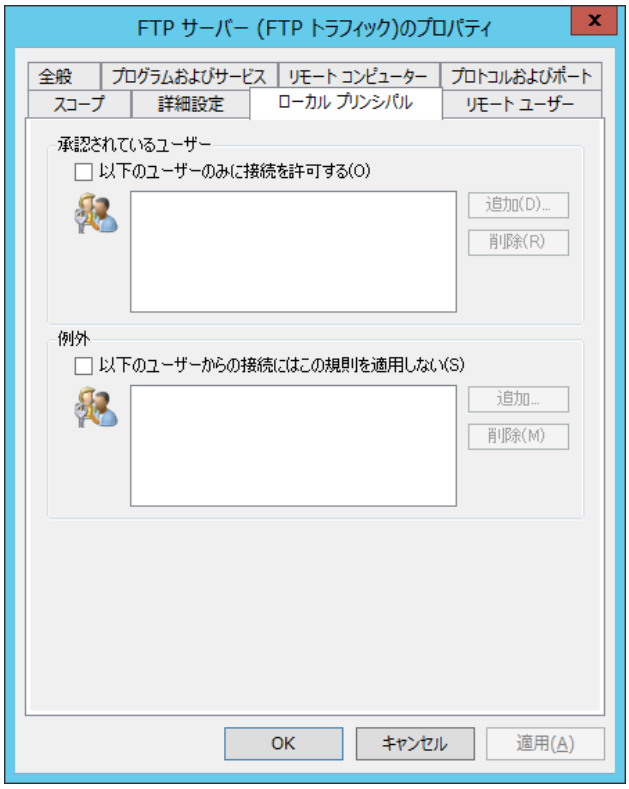


図 2-6-3 承認された FTP ユーザの設定（Windows2012Server）

ログインすることができるアカウントを制限することができる。

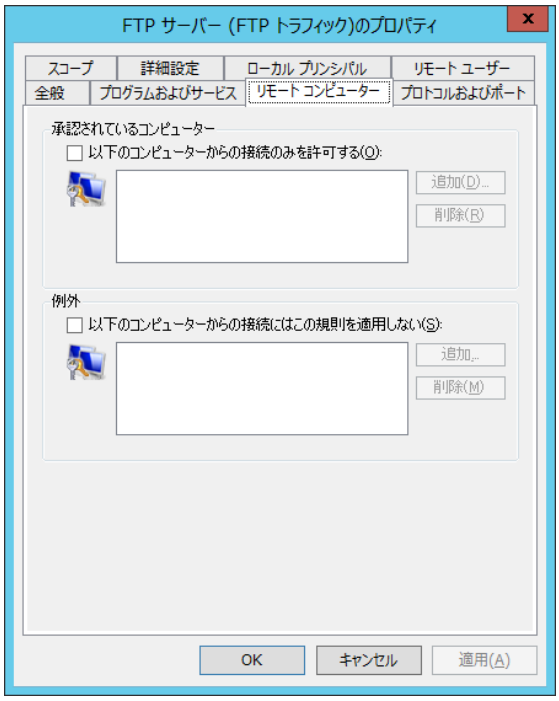


図 2-6-4 承認された FTP アクセス用コンピュータの設定（Windows2012Server）

このサーバにアクセスすることのできるコンピュータを制限することができる。

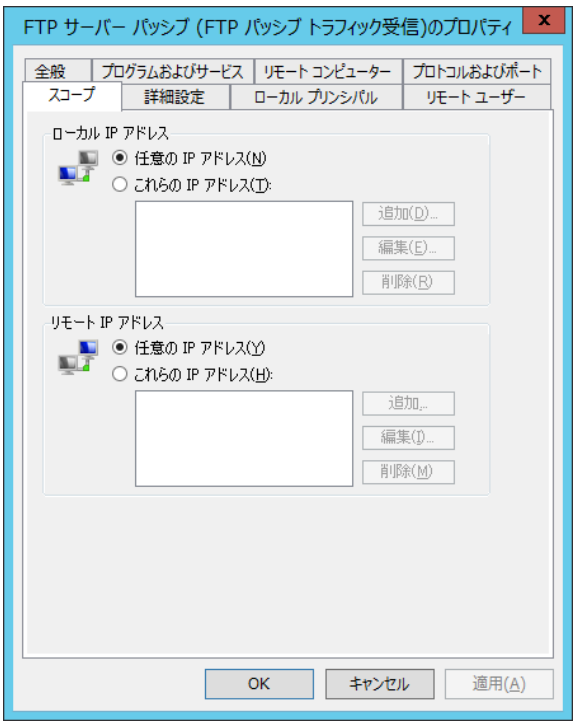


図 2-6-5 承認された FTP アクセス用 IP アドレスの設定（Windows2012Server）

このサーバ自身の IP アドレスと、このサーバにアクセスすることのできる IP アドレスを制限する。

なお、プロキシを介してアクセスする場合には、プロキシのアドレスをここに登録する必要があり、さらに複数のプロキシを用いて負荷分散している場合には、全てのプロキシを登録する必要がある。

ファイアーウォールには、ドメイン、プライベート、パブリックの 3 種類が用意されている。現在の接続方法は、実際の物理的接続にかかわらず、[管理ツール]→[ローカルセキュリティポリシー] →[セキュリティの設定] →[ネットワークリストマネージャポリシー](W7 の「ネットワークと共有センター」に相当)の設定画面により選択されている接続方法である。現在選択されている接続方法は、「セキュリティが強化された Windows ファイアーウォール」の監視画面で確認することができる。現在選択されている接続方法に対応したファイアーウォールの設定が、サーバの動作に影響する。この設定で選択を行っても、「識別されていないネットワーク：ネットワークに問題が発生したか識別可能な特性が不足していることが原因で、識別できないネットワークです」と表示されることがある。

設定対象となる FTP 受信に関しては、

[1]FTP サーバ（FTP トラフィック）

[2]FTP サーバセキュリティ（FTP　SSL トラフィック）

[3]FTP サーバパッシブ（FTP パッシブトラフィック）

の３種類がある。[1]FTP トラフィックだけ許可され、[3]FTP サーバパッシブトラフィックが禁止されている状態では、FTP 接続はできるが、LS コマンド(ファイル一覧表示の要求)などで応答が来ない。

⑥サーバ側のセキュリティに関する小結

Windows2012Server への移行に際して、各種設定内容の基本は従前 OS と変わらないが、設定に使用する操作画面や、設定結果を保存するファイルなどが大きく変化している。

FTP 接続に関しては、セキュリティ確保のために以下の制約条件設定が可能である。

1) 接続するクライアント・マシンの IP アドレス
   - IIS のサーバ全般のプロパティ設定における「IP アドレスとドメインの制限」の機能設定で「拒否」を指定しておき、特定の IP アドレスだけを例外的に許可することにより、アクセスできるマシンを制限することができる。この設定は、配下の FTP サイトの各ディレクトリに関する個別の設定のデフォルト値となる。
   - 必要であれば、個別の FTP サイトに許可する IP アドレスを増補することができる。
   - ファイアーウォールで、FTP 着信のスコープとして、クライアントの IP アドレスを制限することができる。
2) 接続するアカウント
   - IIS のサーバ全般のプロパティ設定で、「アカウントの認証規則」によりユーザを制限することができる。
   - ファイアーウォールの FTP 着信で、リモートユーザを制限することができる。
   - Windows のディレクトリまたはファイル毎のセキュリティ設定で、ユーザを制限することができる。
3) その他
   - 最大ファイルサイズを制限することができる。
   - アクセスすることができるファイルの種類（拡張子）を制限することができる。

⑦FFFTP

サーバの管理に使用するクライアント側マシンで FTP の接続確認や、ディレクトリ一覧を確認するだけであれば、コマンドラインからの FTP コマンドを使用するのが便利である。多くのファイルを一括して転送する場合には、エクスプローラや FFFTP を使用するのが便利である。エクスプローラでは、大量のファイルを送付中に途中でエラーが生じた場合に、以後の動作がキャンセルされてしまうため、最初からやり直す必要が生じる。

FFFTP においては、メニューの[接続]以下で、サーバ毎、ないし同一サーバ上のプロジェクト／サイト毎に接続の単位を作成する。ファイルの送受信に使用するクライアント（ミラーサーバ）との間にファイアーウォール（プロキシ・サーバ）が存在する場合には、メニューの[オプション]以下でファイアーウォールの設定を行った上で、個別の接続単位に関してファイアーウォールを使用するように設定する。

[例]プロキシ・サーバのクライアント側からの IP アドレスを<ip1>、プロキシ・サーバから

見た WEB サーバの IP アドレスを<ip2>とすると、ftp コマンドでこのサーバにアクセスする場合には、cmd.exe、PowerShell.exe 等のコマンドラインツールで

```
open <ip1>
user ユーザ名@<ip2>
pass WEB サーバにおけるこのユーザのパスワード
```

とタイプする。同じ接続条件を FFFTP に指示するためには、図 2-6-6～8 に示した操作画面での設定を行う。

⑧まとめ

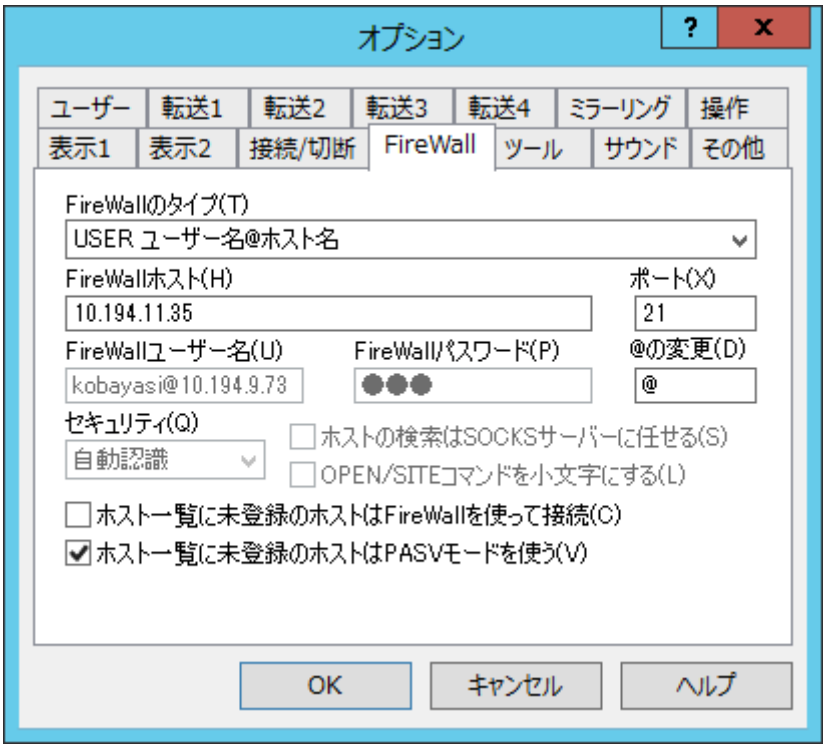


図 2-6-6 ファイアーウォールの設定（FFFTP）

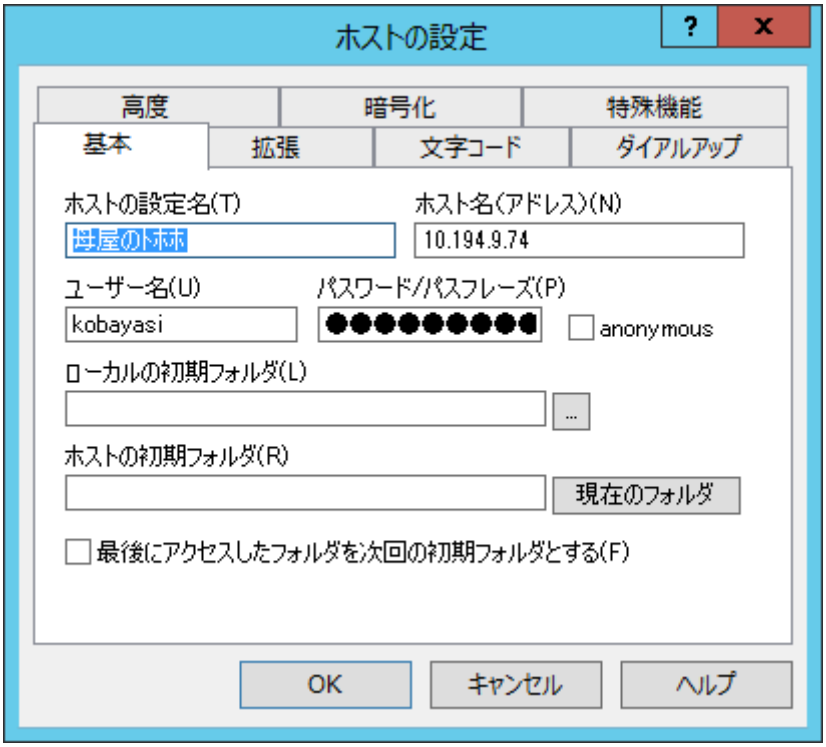


図 2-6-7 アクセス先サーバの設定（FFFTP）

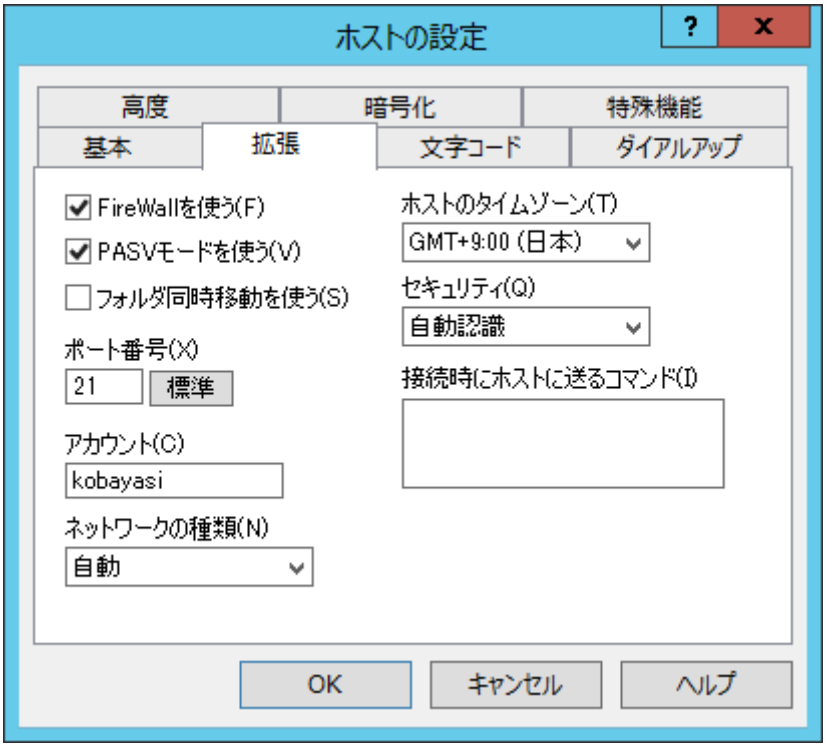


図 2-6-8 PASV モードの設定（Windows2012Server）

（４）データベース（SQL サーバ）

①データベースの歴史と展望

現在「データベース」と呼ばれるものは、単なるデータではなく、動的に維持更新するようなシステムであって、メモリに制約されずに大量のデータをファイルで管理するものである。基本的には突然の電源遮断やシステム・ダウン、あるいはバグ障害が生じても、複数のテーブルが一体に連携したデータの部分的変更により整合性が失われないようなアルゴリズム（トランザクション、コミット、ロールバック）が用意されている。

景観シミュレーション・システム（1993～）において使用している景観データベースは、主に検索機能の実現を目的としてゼロベースで開発したプログラムである（巻末文献 11、建築研究資料 92)。入力用のエディタを用いて三次元データ本体に加えて、様々な検索のための補足情報を登録し、入力されたデータベースは、COM.TXT というテキストファイルに保管した。データベースの入力は、editor.exe という入力用の Windows 実行形式を用いて行い、検索は三種類の実行形式（yuu.exe:優良景観事例、zai.exe:景観材料、kou.exe:景観構成要素）を用いて検索した。データベースはこの段階では小規模であったため、editor.exe の動作中は、全てメモリ上に展開され、入力が終了した時点で、外部ファイルを更新・保存する。ファイル保存中にシステム・ダウンが生じた場合に過去の累積データが失われないように、バックアップを自動的に作成するようにした。よって、損失はメモリ上に全ロードされたデータに対する更新部分のみである。

まちづくり・コミュニケーション・システムの開発(2001～)に際しては、このデータベースを WEB サーバ上で運用するシステムを実現すると共に、データベースの構成（スキーマ）をプログラムで固定的に扱うのではなく、DEF.CSV という外部ファイルで定義することにより、様々な形式のデータを扱うようにした（巻末文献 21、国総研資料 134)。この処理系においては、MSDE という無償で再配布することができる Microsoft 社が提供するデータベースエンジンを使用し、上記の補足情報（属性情報）を管理すると共に、三次元データ自体は、固定的なファイルとして、サーバ上のディレクトリに置いて管理した。このシステムにおいては、三次元形状を記述したデータは、アップロードされたファイルのまま蓄積し、その内部を図形的に解析してテーブルに展開するような処理は行っていない。

２－７、３－６で解説する VC-4D(三次元データ保管庫、2012～)においては、三次元データ自体も、頂点座標、線、面、立体、色彩等の要素に分解する。これにより、異なるデータ形式への変換を可能にしている。

データベースエンジンとして使用した SQL サーバにおいては、複数の異なる形式のデータの集まりを一つのレコードとし、任意の数のレコードの集まりをテーブルとして管理している。C 言語風に表現すれば、構造体の配列であり、構造体の定義がスキーマである。

このテーブルの構造は、Lotus, Excel のような表計算ソフトにおける表と類似しているが、列の数がスキーマで予め定義された数に限定されており、また集計式等を項目の中に埋め込むことはできない。

レコードの構成要素は、データそのものであっても、別のテーブルの要素への参照であっても良い（リレーショナルデータベース)。例えば、ある物件のテーブルに所在地都道府県の項目がある時に、都道府県のテーブルが別途用意してあれば、全ての物件に関して文字列で所在地を記入する代わりに、都道府県のテーブルの ID を指示するだけで済ませることにより、メモリやファイルサイズを節約することができる。

このような一連のテーブルを束ねたものを「データベース」として、一つのファイル(MSSQL の場合、拡張子.mdf)として動的に管理している。ファイルを元にデータベースエンジンを使用することにより、停電による全データ消失を防ぐことができる。

但し、例えばある頂点座標を定義した上で、その ID を用いて面を定義するような処理を行っている途中でエラーが生じた場合には、例えば参照されない頂点座標や、参照先のない面のデータが生成する恐れがある。そこで、このような一連の処理を TRANSACTION として束ね、一連の処理が終了するまえに不測の中断が生じた場合には、外部ファイルとして保存されるデータベースに反映しないように保留することができる。TRANSACTION が終了した段階で、問題が無ければ COMMIT することにより結果を確定することができ、また問題があれば ROLLBACK することにより処理前の状態に戻すことができる。これにより、たとえ 1 件のデータ投稿が不成立に終わっても、データベース全体の一貫性が失われないように管理することができる。

②利用可能なデータベースエンジン

MSSQL で、無償で再配布することのできるデータベースエンジンは、2001 年頃の MSDE から、SQL EXPRESS と名称が変更され、2005, 2008, 2012 等のバージョンに改良されている。

プログラム開発途上にあっては、データベースの内容を例えば表形式で表示し訂正したり、各種のサンプルデータを手入力したりする道具があると便利である。MSDE の頃には、このような道具として、Enterprise Manager というアプリケーションが提供されていた。その後、この道具は、Development Studio という名称になり、開発環境と一体で提供されるようになった。

開発環境 VS2005 を搭載した Windows8.1 にセットアップされている、上記の条件に該当する SQL データベースは表 2-6-0 の通りである。

表 2-6-1　無償で利用可能な MSSQL サーバ(2015 年時点)

2005
2005 Mobile [JPN] Developer Tools
2008 R2 管理オブジェクト
2008 セットアップサポートファイル
2012 (64 ビット)
2012 Express LocalDB
2012 Native Client

データベースへの操作は、sqlcmd.exe(isql.exe, osql.exe の後継)によって SQL 文をコマンドラインで実行することができる。コマンドライン(cmd.exe または powershell.exe)の画面から sqlcmd とタイプすることにより起動できる。sqlcmd は ODBC インターフェースを

用いてデータベースエンジンにアクセスするため、ODBC に使用するデータベースが登録されている必要がある。

Windows2012 Server (standard edition)をサーバの OS として、この上に、現時点で最新の無償 SQL サーバをダウンロードし、セットアップをテストした結果は、以下の通りである。データファイルとして同じデータベース(拡張子.mdf)を利用するために、使用できるエンジン、接続するためのプロバイダ、アクセス方法などが選択可能である。過去 20 年の流れで見ると、多様な設定やアクセス方法が可能となった一方、初期導入には手間と時間がかかるようになった。

１）2012 Express LocalDB (ローカル DB)

LocalDB は、従来の MSSQL サーバがサービスとして動作するのとは異なり、実行形式（インスタンス）の形で動作する。セットアップした後に、コマンドラインで動作する sqllocaldb.exe を起動し、コマンドを入力することにより、様々な設定を行う。

＞create 名称 : 名称のインスタンスを作成する。デフォルトで、v.11 というインスタンスが一つ起動しており、複数のインスタンスを作成することができる。

＞start 名称 : 名称で指定したインスタンスを開始する。

＞stop 名称 : 名称で指定したインスタンスを終了する。

＞delete 名称 : 名称で指定したインスタンスを削除する。

＞info : 現在作成されているインスタンスの一覧を表示する

＞info インスタンス名称 : 名称で指定したインスタンスの状態を表示する

このインスタンスが開始され実行中である場合に、リストの末尾に「インスタンス　パイプ名」として表示される「名前付きパイプ」の名前がアクセスに必要である。このパイプ名称は、例えば

「np:¥¥.¥pipe¥LOCALDB#F3A1FC0B¥tsql¥query」

といった比較的長い名称である。

開始している実行されているインスタンスについて、コマンドラインツール sqlcmd.exe を起動し、

＞sqlcmd –S パイプ名称 –E

により、ログインが成立すると、１＞のプロンプトでコマンドラインから SQL 文を受け付け実行することができるようになり、以後データベースの作成や検索・更新など各種の操作を行うことができる。

LocalDB のセットアップは３６MB 程度のサイズであり、ダウンロードは短時間で終わり、セットアップ全体は数分以内で完了する。

なお、sqlcmd は、ODBC ドライバを使用しているため、LocalDB にアクセスするためには、事前の登録が必要である。「コントロールパネル」から（[全てのコントロールパネル項目］－）［管理ツール］－［データソース(ODBC)］の順に選択し、LocalDB を［追加］登録する。

２）SQL2012 とマネジメント・スタジオ

開発・デバッグ用としてデータベースの内容の確認や、バックアップ条件等各種の設定を試行錯誤的に行うための Management Studio の機能を含む SQL 2012 のダウンロード(http プロトコル)とセットアップには、相当の時間（2015 年 7 月時点、筆者の作業環境では約半日）を要した。そのほとんどは、コマンドで指示した処理の終了待ちである。

マネジメント・スタジオ(Management Studio)は Enterprise Manager の後継で、グラフィカルな画面でデータベースの管理やデータの操作を行うことができる。これが使用する.NET Framework3.5 は、共通言語基盤(CLI, common language Infra. JIS, ISO 等として標準化されている）として、各種言語のソースコードからコンパイルした結果を中間言語として処理する。CLI は、基本クラスライブラリ（BCL）と、仮想実行システム（VES）から成っている。これにより、C#や VB.NET 等のソースコードから共通の中間言語（CIL）を生成し、これを更に共通のコンパイラ(JIT、just in-time コンパイラ)により、実行時にコンパイルすることにより各処理系のネイティブコードにコンパイルして実行する、二段階構成のコンパイラシステムを実現している。

このため、Management Studio を導入するためには、それが依存する.NET Framework の先行導入が要求されており、条件が満たされないと、SQL サーバのセットアップは失敗終了となる。なお、場合によっては、SQL2012 のセットアップの途中で NET Framework のインストールには、「Windows の機能(Windows features)」のダイアログで、「この機能をインストールします。」という選択肢が示される。

.NET Framework は、コントロールパネルの「プログラムと機能」の中に存在し、選択してアンインストールすることができる。また、左側に表示されている「Windows の機能の有効化または無効化」をクリックすると、インストールされている各機能について

有効／無効をセットするためのチェックボックス群が縦に表示される。

＊Windows2012Server の場合には、上記の操作を行おうとすると、「サーバマネージャ」が起動される。左欄でローカルサーバを選択して右画面を下の方にスクロールすると、.NET Framework 4.5 および .NET Framework 4.5 Features(機械翻訳では、Features は通常「機能」と訳される)の項目がある。これは表示のみで、有効／無効を切り替えるインターフェースは見られない。セットアップは失敗と表示されるものの、より上位の.NET Framework 4.5 が動作しているためか、サーバマネージャなどの設定画面は開くことができ、セットアップしたサーバの設定を行うことができる。但し、たとえこれが有効であっても、より古いバージョンの.NET Framework3.5 がセットアップされていなければ、Management Studio をセットアップすることができない。

.NET Framework 3.5 は OS 導入時に自動的にセットアップされていないため、新たに導入する必要がある。このためには、サーバマネージャのダッシュボードで、役割と機能の追加を選択し、役割と機能の追加ウィザードを開く。「機能選択」画面で、NET Framework 3.5 Features の左のチェックボックスにチェックを入れる。サーバが Windows Update（インターネット）に接続できない環境（バリアセグメント等）にある場合、セットアップがプログレスインジケータの表示４０％程度まで進行した所でエラー終了する。この場合、Windows2012Server（セットアップディスク）を DVD ドライブにセットし、例示に従って、E:¥sources¥install.wim を代替サーバとして指定する（タイプ入力）。Install.wim とは、Microsoft Windows Imaging Format(WIM)形式による OS のインストール元である。

インストール媒体（DVD-ROM）を使用する場合には、次の設定コマンドをコマンドラインから管理者権限で実行し、イメージの展開を行う必要がある。

```
Dism /online /enable-feature /featurename:NetFx3 /All /Source:E:¥sources¥sxs /LimitAccess
```

ここで、Dism とは、展開イメージのサービスと管理を行うコマンドである。
/online は、Windows の機能に関する
/enable-Feature は、削除された Windows の機能を復元する
/featurename: は機能の名称　NetFx3 は、NET Framework 3

「機能を有効にしています」の次の行にコマンドラインのプログレスインジケータが表示され、66.2%で長く止まるが、100%まで到達する。なお、ソースの指定が誤っていても、途中まで進行して、エラーで終了する。この処理に約 1 時間。失敗した場合に参照可能な DISM ログファイルは c:¥Windows¥Logs¥DISM¥dism.log にある。

恐らく、Windows Update をインターネットで許可してあれば、この辺の処理は自動的に行われるのであろう。保護された環境下での作業を行う方法は用意されているが容易ではない。成功すると、「操作は正常に終了しました。」と表示。

これにより、セットアップが前に進む。この作業には 1 時間程度を要する。この作業の間、システムはネットワークから受信を断続的に行っているが、CPU 時間はあまり消費していない（１０%未満）。セットアップや試行錯誤の終了後、無効化ないしアンインストールしておいた方が安全である。

SQL サーバのセットアップに際して、既定のインスタンス（名称：MSSQLSERVER）か、あるいは名前付きインスタンス（名称：任意）かを選択するダイアログが表示される。（この点は、SQL Server Express は、常に名前付きインスタンスでインストールされるのと異なる。）名前付きインスタンスで名前を指定しなかった場合には、名前の無い名前付きインスタンスとしてインストールされる。この場合、デフォルトの SQLExpress が自動的に、インスタンス名となる。名前付きの場合には、「サーバ¥インスタンス名」でアクセスする必要がある（例えば、「.¥SQLExpress」）。

インスタンスとはメモリ上にロードされた実行形式のことである。同一のプログラムの複数個がメモリ上の別の場所にロードされた場合には、異なるインスタンスとなる。

認証モードについて、Windows 認証か、混合認証かを選択できるダイアログが表示される。Windows 認証の場合にはパスワードはないが、混合認証の場合には s a にパスワード設定が求められる。以前のバージョンとは異なり、SQL Server 2012 においては、パスワード無し、あるいは単純なパスワードはエラーとなる。覚えやすい複雑なパスワードを設定するのが便利である。ここで「複雑」とは、アルファベットと数字と記号が含まれていることを示す。よって、「n=3」は複雑なパスワードとして評価される。

③ コマンドラインを用いた、ＳＱＬサーバの検証

データベース・システムがセットアップされた後に、簡単にコマンドラインから接続して検証することができる。

`sqlcmd –E` で認証プロセスなしにアクセスできる。

`sqlcmd –U sa` でユーザを sa とすると、次行のプロンプトでパスワードを入力し、アク

セスできる。パスワードが設定されていなければ、改行するだけでよい。

データベース名を指定する場合には、-S でデータベース名を指定する

sqlcmd –S （SQL サーバ名） -E

SQL サーバ名は、セットアップ時に既定のインスタンスとして指定した場合には、データベース名のみでよく、またデータベース名自体を省略してもよい。sqlcmd(改行)と入力するだけで、既定のデータベースが起動する。

一般には、マシン名￥SQL サーバ名 の形式で SQL サーバ名を指定する。

起動が成功すると、1> のように SQL の入力を求めるプロンプトが表示される。

更に、この前にプレフィクスを付けることにより、名前付きパイプ、IP などの接続方法を指定することができる。

ローカルマシンの場合には、マシン名の部分に当該マシン名を入力する代わりに、”.”、“(loal)”、”localhost”等の一般名とすることができる。

sqlcmd -? で、各種のパラメータ設定の一覧を表示することができる。

```
1 >select @@version
2 >go
```

で、システムのバージョンを表示することができる。

```
1>exit
```

で終了する。

このサーバ上にデータベースを一つ構築するためには、

```
1>CREATE DATABASE 名称
2>GO
```

と入力する。データベースの名称や場所等を引数で詳細に指定することができるが、省略することもでき、その場合には同一の名称のファイルが標準の場所に作成される。

このデータベースを選択して、内容を表示し変更するためには、

```
1>USE 名称
2>GO
```

と入力する。

作成されているデータベースを削除するためには、

```
1>DROP DATABASE 名称
2>GO
```

と入力する。このデータベースが指定されている状態ではエラーとなる。

現在選択されているデータベースのテーブルを全表示するためには、

```
1>SELECT * FROM テーブル名
2>GO
```

と入力する。

システムデータベースとして最初から存在している master と同一名称のデータベース

を作成することはできない。また、この master データベースを削除することはできない。

SQL サーバに接続する際に、データベース名を指定しなかった場合には、接続が確立した段階で、master データベースが選択されている。新たなデータベースを作成した場合には、対応する保存ファイルが作成されると共に、これらの情報が master データベースに登録されている。

```
1>SELECT name FROM sys.databases
2>GO
```

というコマンドにより、現在のデータベースの一覧を表示することができる。この一覧の中には、master 自身も含まれている。

SQL サーバへの接続のプロトコルには、共有メモリ、TCP/IP、名前付きパイプ、VIA があり、SQL Server Configuration Manager によってそれぞれの有効／無効を設定することができる。

一方、接続するクライアントの側では、SQL サーバ名にプレフィクスを付けることにより、プロトコルを指定することができる。

1) 名前付きパイプ

np:¥¥<コンピュータ名>¥<パイプ名>

2) TCP/IP

tcp:<コンピュータ名>,<ポート番号>

3) 共有メモリ（主にトラブルシューティングに用いる、ローカルな接続）

lpc:<コンピュータ名>¥<インスタンス名>

4) VIA プロトコル（仮想インターフェースアダプタ）

ネットワークインターフェースカード番号（NIC 番号）とポート番号を指定する

via:<SQL サーバ名> [¥<インスタンス名>],<NIC 番号>:<ポート番号>

NIC 番号が省略された場合には、1433 番がデフォルト番号となる。

複数のインスタンスがある場合に推奨されない。

```
現在の接続方法を確認するためには、
SELECT net_transport FROM sys.dm_exec_connections WHERE session_id=@SPID;
```

基本的には、コマンドラインだけで複雑な操作を実現することができ、またそのような手順をスクリプトとして保存しておき、異なるデータベースに対して繰り返して適用するようなことも可能であるが、途中でエラーが生じた時の処理結果のログやメッセージを作成し、中断終了の分岐処理などを記述することは困難であるため、より高度なスクリプト言語を用いて処理を記述するのが便利である。

データベース作成処理の内容は、プロジェクト毎のデータベースの構築と、初期状態（空）のテーブル群の作成である。

まちづくり・コミュニケーション・システムの場合にも、VBScript を用いて、データベースを構築する処理を記述しており、その過程での各処理の成否については、ログファイ

ルを出力しているので、結果を確認するのに便利である。

バックアップ方法等は、SQL サーバにより異なるため、以前の Enterprise Manager や、上記の Management Studio などの専用の設定ツールを用いて作業を行うのが便利である。これらのツールを用いて、データベースの各テーブルの内容を閲覧し修正することもできるが、大量の一括処理を行うことは実用的ではない。

このようなツールをセットアップせずとも、データベースエンジンと、実用システムの一部として作成してある専用のデータベース構築・削除ツールがあればシステムを構築し運用し廃止する作業は行うことができる。しかしながら、システムを運用する中で、悪意のある攻撃、システムのバグによる不適切なデータの生成などの想定外の状態が生じる場合があり、その検査と緊急対応的な作業ツールとしては便利である。

一方、このようなツールが存在することは、SE 等の立場の人達がまちづくりに関する意見（テキスト）を改ざんしたり、あるいは一部の意見だけを削除したり、といった不正を容易に行えるようにするような負の側面も有している。セットアップや試行錯誤の終了後、無効化ないしアンインストールしておいた方が安全である。

まちづくりコミュニケーション・システム(2001)においては、プログラム的要素を含む意見投稿などのハッカー的な攻撃に関しては、SE や管理者（いわば事務方）による主観的な管理ではなく、審査システム（いわば外部査読者）を通じて不合格判定とし、公開される段階に移行しない仕組とした点に特徴がある。技術面を支えるべきシステム管理者等による主観的な評価だけで、反対意見を封じこめるようなデータ介入は行えない事が望ましい。

④地区別 WEB サイトのためのＳＱＬサーバの設定と検証

１）アクセス文字列

従来の VB スクリプトにおける古いスタイルのデータベースアクセス：

```
SET DB=CreateObject(“ADODB.Connection”)
Cstrg=”Provider=MSDASQL;DRIVER:sql server; SERVER=(local);
DATABASE=”&dbname&”; UID=”&dbuser”;PWD=”&dbpass
DB.Open Cstrg
```

修正後のスクリプトにおけるデータベースアクセス：

```
SET DB=CreateObject(“ADODB Connection”);
Cstrg=”Driver={SQL Server}; server=; uid=”&dbuser&”; pwd=”&dbpass&”;
DB.Open Cstrg
```

なお、古いスタイルは、勧奨されていないが、接続はまだ可能である。

ここで、ADODB.Connection とは、Active Data Object を接続プログラムの組込み方法として用いることを示す。 Provider は、接続のために使用するプログラムであり、MSDASQL.dll を使用している。Driver は、ODBC(Open Data Base Connection)の中で使用するドライバである。どのドライバが利用可能であるかは、ODBC 設定ツールである odbcad32 つまり OpenDataBaseConnectionAdministrator32bit 形式を用いて、確認し追

加することができる。コマンドライン窓から odbcad32 とタイプすることにより起動する。図の例では、SQL Native Client、SQL Server、SQL Server Native Client 11.0 の 3 種類が、ドライバとしてセットアップされていることがわかる。Server は、アクセスするサーバであり、記述方法はデータベースのセットアップ方法に依存している。

(local)は、古いスタイルで、コンピュータ名¥サーバ名が、完全な記述である。ローカルコンピュータの場合には、.¥サーバ名 の形で省略することができる。

サーバ名が省略されると、Native Client サーバへのアクセスが試みられ、失敗するとタイムアウトする。

利用可能なサーバは、Microsoft SQL Server Management Studio を起動して、接続する際に参照を選択すると、一覧として見ることができる。

なお、ODBC データソースアドミニストレータを用いて、DSN（データソース名）を定義し、”DSN=・・・”という接続文字列を用いて SQL サーバにログインする方法も可能であるが、セットアップ作業が一工程増えることとなるため、今回は現実的・実用的観点からこのような理念的対応を棄却した。何故ならば、そのような互換性は過去に遡ったプログラムの修正により将来にわたるプログラムの互換性を保証する、というアプローチであり、現実的には過去に遡る必要はなく、また将来 SQL サーバが供給され続ける保証もないからである。

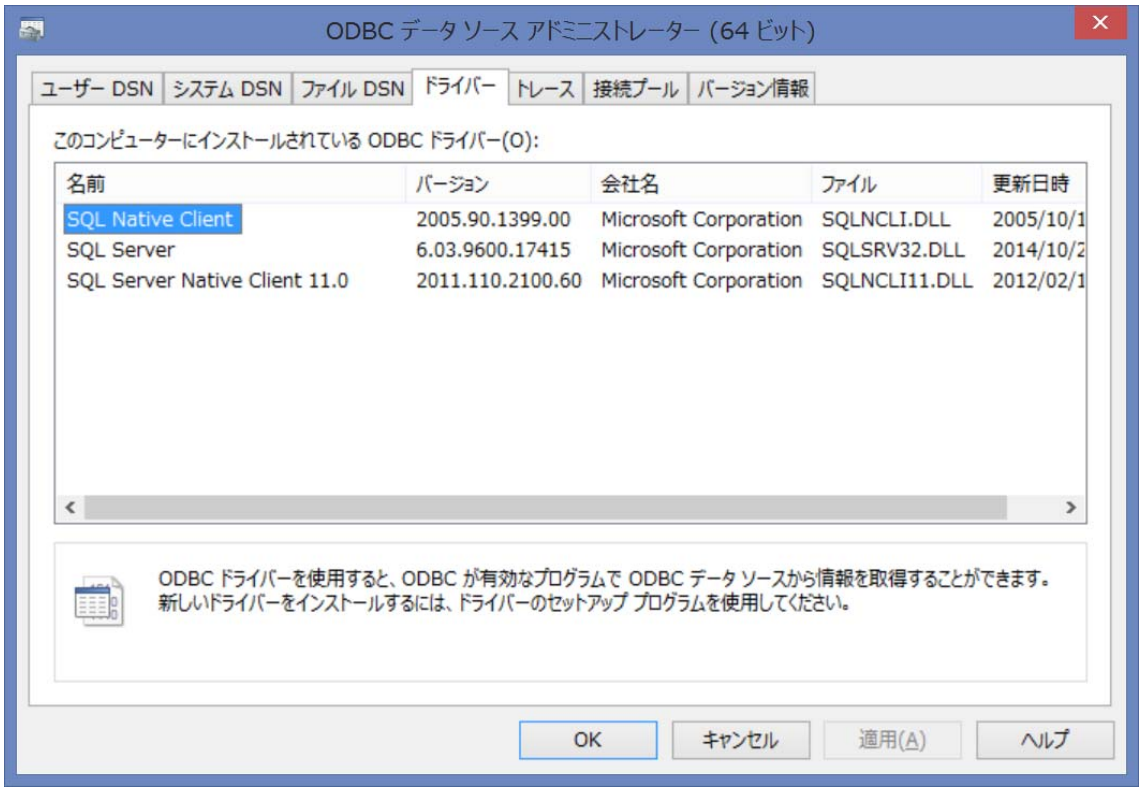


図 2-6-9 データベースドライバの選択と設定（Windows2012Server）

２）asp のエラーメッセージの表示

asp は、サーバ側のプログラムを記述する方法であり、本処理系においては VB スクリプトを用いている。html 形式のコンテンツの中に、<% ・・・%>で囲んだ中に、BASIC 言語による記述を行う。

IIS の設定で、奥尻サイトの asp のプロパティを開き、デバッグのプロパティの下にある、ブラウザへのエラー送信を、false から true に変更する。

データベース名は、ダイアログから入力されているものを使用している。新たなデータベースを作成する際には、この名称のデータベースはまだ存在していない筈である。その場合であっても、DB.Open は実行され、master データベースが選択された状態になる。master には、ユーザが作成(create)したデータベースが登録されている。

スクリプトの実行がコンパイル失敗により E500 となる場合（ブラウザ側での表示)、引用符”の不適切な使用が原因である場合が、経験的には多い。

コンパイルが成功し、スクリプトが実行された後に発生したエラーの個数は、DB.Errors.Number メンバ変数によって知ることができる。但し、Open に際しては、データベース名が指定されていないか、指定された名称のデータベースが無い場合であって、master データベースが代わりに開かれたような場合の警告、および言語を日本語に変更した警告も DB.Errors.Number としてカウントされる。よって、警告を除いた実害のあるエラー数を数えるためには、全てのエラーにアクセスして、内容を調べる必要がある。

リスト 2-6-1 クラシック asp による SQL コマンドエラーの表示関数

```
Function sdb_error(DB)
                dim count
                dim nerr
                count = 0
                Response.Write "【エラーチェック開始】<br>"
        For Each errLoop In DB.Errors
            If 0 <> errLoop.Number Then
              count = count + 1
            End If
            strError = "Error #" & errLoop.Number & "<br>" & _
                "     " & errLoop.Description & "<br>" & _
                "     (Source:" & errLoop.Source & ")" & "<br>" & _
                "     (SQL State:" & errLoop.SQLState & ")" & "<br>" & _
                "     (NativeError:" & errLoop.NativeError & ")" & "<br>"
            If errLoop.HelpFile = "" Then
                strError = strError & _
                    "     No Help file available" & _
                    "<br><br>"
            Else
                strError = strError & _
                    "     (HelpFile:" & errLoop.HelpFile & ")" & "<br>" & _
                    "     (HelpContext:" & errLoop.HelpContext & ")" & _
                    "<br><br>"
            End If

            Response.Write("<p>" & strError & "</p>")
        Next
        sdb_error = count
        DB.Errors.Clear
        Response.Write "【エラーチェック終了】<br>"
End Function
```

例えばリスト 2-6-1 に示した sdb_error 関数によってエラーの原因を表示させることができる。このようなエラー表示機能は、サーバの更新に伴う障害の原因を特定するツールとしては有効であるが、設定が終了した後には削除するか不活性化しておくことにより、セキュリティ上の弱点とならないように対策しておく必要がある。

⑤ 掲示板機能のためのテーブルの構築

個々の地区別のコミュニケーション・サイトのための SQL データベース（空）を作成した上で、その中に事業別の初期状態の各種テーブルを作成する。テーブル構成はスキーマである def.csv ファイルによる定義に従う。この作業はコマンドラインや Management Studio などの管理ツールから行なうこともできるが、手間を要する。このため「まちづくり・コミュニケーション・システム」のために開発したデータベースとテーブルの自動作成のためのスクリプトが、新しい OS と SQL サーバにおいても使用できれば便利である。

Windows2000Server 上で開発した初版については文献 21（pp.42～56）で解説した。その後、Windows2003Server に移植した。この方法をそのまま Windows2012Server 上で実行しても旧 OS 上のように正しくテーブルを構築することができない。そこで、奥尻サイトに含まれる掲示板機能を例として、デバッグ情報の取得・参照方法と、それを用いて修正したプログラム（スクリプト）や作業手順を以下に解説する。

１）テーブルの自動構築のためのスクリプト

[1] アクセス情報等を記録するテーブル群の作成

WEB サイトを構成する各ページ(.asp)へのアクセスを SQL サーバに記録する機能である。必要とするテーブル群を作成するためには、WEB ブラウザでアクセスしたサーバ上の hdbmk_counter.asp を起動すると、ここから呼び出されるインクルードファイル tdbmk_database.inc で定義された関数が、データベースを作成し、tdbmk_counter.inc で定義された関数がアクセスカウンタに必要なテーブルを作成する。この単純なテーブルは、ページのファイル名、累計ヒット数、最後のアクセス、最後の更新日を記録している。作成するデータベース名、ローカルファイル名、ユーザ名、パスワードは tdbm,_database.inc のプログラム中に直接に記述しているため、運用するシステムに合わせて書き換える。

この構築過程の途中結果は、成否にかかわらず全てテキスト形式で記録され、終了時点でブラウザ画面に表示される。 更に、エラーが生じた場合には、Windows 配下の LogFiles の中に、W3SVC のログとしても記録されるため、デバッグの手がかりを得ることができる。

データベースとテーブルの作成が完了した後に、新たなページを(.asp)作成し最下行に

<--#include FILE=”include/t_counter2.inc” -->

というインクルードでスクリプトを追加すると、このページに最初にアクセスした時点でこのページ（ファイル名）が上記のテーブルに追加されると共に、カウンター１にセットされる。カウンターの値はアクセスの度にインクリメントされ、ページの中に表示される。

カウンターはデータベースが動いているかどうかの簡便なチェックとしても有効である。

[2] 意見投稿と審査のためのテーブル群の作成

ユーザからの意見や資料の投稿を受け付け、審査員によるオンライン審査の上で公開する機能である。この機能において作成するテーブル構成は、スキーマに相当する def.csv 形式のファイルによって目的に応じて自由に定義できる。テーブル群の作成は、WEB ブラウザからアクセスしたデータベース作成ページ(HP_dbmk.asp)でパラメータを設定し実行ボタンを操作することにより行なう。このスクリプトは、スキーマで定義された投稿意見のテーブル以外に、審査の進行状況や各審査員への依頼数を管理するテーブルも作成する。

移植後のデータベース名、アカウントなどは、tdb_user.asp の中で定義するため、移植先のシステムに合わせて修正する。

テーブル構成を定義するスキーマである def.csv 形式のファイルの中では、データを配置するローカルアドレスも定義しているため、移植先に合わせて書き換える必要がある。なお、def.csv 形式のファイルは複数個システム上に存在していてもよく、名称は、def.csv とは異なっていてもよい。よって移植前の def.csv を別名または別場所に残しておくことができる。使用するファイルは、VB スクリプトが参照する asp-cfg.txt ファイルと、tehaishi サービスが参照する cfg.txt によって、フルパスで特定する。

asp ファイルに含まれる VB スクリプトをデバッグするためには、「idebug=1 」といコマンドでデバッグ用のフラグを立て、豊富なメッセージを出力させる。必要であればバッグ用のコマンドを追加すると共に、エラー発生箇所付近に、Response.End コマンドを追加してそこで処理を中断することにより、問題箇所を特定する。

上記[1]と[2]の作業を通じて、結論的に W2000Server 上で開発し、その後 W2003Server に移植し稼働していた全ての機能を、W2012Server 上に移植することができた。なお、[1]と[2]を同じデータベース上に共存させることも別のデータベースとすることもできる。同名のデータベース上に共存させるためには、[2]→[1]の順で行なう必要があるため、デバッグを伴う移植に際しては単純な[1]のテストを通過した後、一度削除して[2]に進むのが速い。

２）SQL2012 への移行における修正点と留意事項

Windows2003Server 以前の OS におけるテーブル作成のためのスクリプトや作業手順を W2012Server＋SQL2012 に移行するための修正点と留意事項を、以下にまとめる。

[1] デバッグのため、idebug という変数に 1 を代入しておくと、デバッグモードで処理が行われ、詳細なエラーをクライアント側に表示するようになる。プログラムからメッセージを作成し、これをクライアント側に返送する方法は、IIS の仕様変更等にかかわらず有効であるため、システムの載せ替えが楽である。しかしながら、審査機能付き掲示板系のデータベース作成のコールバック（form の action=”tdbmk_main.asp”）におけるエントリポイント tdbmake_main 関数から呼び出す tdbmk_database 関数は、インクルード(tdbmk_database.inc)の中で定義されている。tdbmake_main 関数では idebug 変数に 1 を代入しているにも関わらず、tdbmk_database 関数の中で idebug により処理を切り替えているケースでは、多くの場合、idebug=0 の側に分岐している。つまりインクルードにより参照したファイルの VB スクリプトにおけるグローバル変数の外縁は不明確である。よって、

各種関数には、idebug を引数として明示的に渡すのが好ましい。そこで、tdbmk_database 関数に引数を一つ追加し、呼び出し元から idebug を引数として与え、呼び出し先で引数 idebug として受け取る方法により、この不整合を解決した。

[2] クライアント側にエラーメッセージを返送する方法として、Response.Write 関数で直接メッセージを作成する方法、tAdd_Serverlog(文字列)で、サーバログにエラー情報を追記する方法、tMkKekkaOutcom_html(文字列)により、最終的に作成する処理終了後の一次的な表示頁にエラー情報を追記する方法が用意されている。後２者の方法では、最後に転送したメッセージのみが表示されるため、途中で生じたエラーに関する情報は掻き消されている。このような諸点に関して、設定手続きのためのプログラムは改良の余地がある。

[3] 上記 tdb_makedatbase.inc および tdbmk_database.asp の中で SQL コマンド文字列を発行してデータベースの構築している。コマンド文字列の中で指定する mdf ファイル(データベースを格納する)のサイズとして従来、固定的に 1 MB を指定していたが、SQL2012 では、最低５MB を求めており、これより小さな値を指定すると、エラー終了となる（データベースは作成されない)。その際のエラーコードが、データベースが既に存在する場合と同じであるため、スクリプトからは「その名前のデータベースは既に存在」という表現のエラーメッセージが表示されるが、「データベース指定における数値の過小」とすべきである。

エラーコード番号ではなくメッセージ文字列を返すのは親切である。しかし、上記のようにコードの意味がバージョンにより変化し混乱する場合もあるため、固定的な（古い）変換表で文字列に変換せずに、更新後の最新システムにおけるメッセージを文字列として取得した上でそれを返送する方が、SQL や IIS の仕様変更に対して柔軟に対応ができる。

[4] データベースを作成するディレクトリとして、存在しない場所の名称を入力した場合には、ルートにデータベースが作成され、成功裡に終了するので注意が必要である。

データベースが既に存在する場合、内容が上書きされるように解説されているが、実際にはエラー終了する。よって、予め前述の方法で、カウンターやアクセスログのためのデータベースがプロジェクト名称を用いて作成されている場合には、それに各種のテーブルを追加する処理には進まない。

[5] IIS8.0 においては、asp スクリプトの中でのファイル関連処理において、親ディレクトリへのアクセスを規定（初期状態）で禁止しており、本処理系ではこれを設定で許可する必要がある。しかし、この設定は、IIS を再起動するまで反映されない。

親ディレクトリ参照ができるようにすることにより、プログラムの可搬性を高めることができる。実際には、まちづくりコミュニケーション・サイトのそれぞれの動作を定義する DEF.CSV ファイルの中で、ディレクトリを指定する際に、「../」から始まる相対アドレスを用いることにより、サーバの更新等に伴う修正箇所を大幅に減らすことができる。

[6] エラーメッセージの表示方法、エラーログの記述内容に関して、全般に、エラーメッセージは攻撃者に対して有用な情報を提供するリスクが大きいため、クライアント側には返送せず、サーバ上のログファイルとしてのみ記録を残すような方法が勧奨されるようにな

った。このため、OS の更新や SQL データベースの変更等に伴う作業において、エラーに関する情報は取得しにくくなっている。

現在では、障害無く確実に動作するシステムを完成させることよりもむしろ、悪意ある攻撃に対する安全性が問われている。セキュリティを高めるために屋上屋を重ねるような対応もありうるが、テストに使用する管理機能などは、設定終了後には、たとえ SE であっても不正アクセスできないように封鎖ないし削除しておくのが単純な回避策となる。

⑥ basp21 について

2001 年に本処理系を運用開始した段階で、サーバ側の ASP（2015 年現在では Classic ASP と呼ばれている）におけるファイル処理機能を追加するプラグインとして、有用であった。同じ実行形式は、Windows2012 上でも動作することを確認したが、３２ビットの実行形式を、６４ビットの OS 上で実行するために、IIS のアプリケーション・プールにおいて、３２ビットのプラグインの実行を許可するような設定を行わなければ正常に動作しない。

⑦ 巡回サービス

投稿状況をチェックし、新規投稿があれば査読委員に査読を依頼し、締切が迫った時に査読結果が届いていなければ督促を行い、査読結果が集まれば集計して掲載の合否を決定する、tehaishi という名称のサービスである。

Windows2012Server においても、サービスを登録して起動することができる。コマンドラインで「.¥tehaishiserviccconfig」とタイプして TehaishiServiceConfig.exe を起動すると、同じディレクトリに存在する cfg.txt で所在場所をフルパス指定した、事業別の設定を記述した def.csv の定義に従って、サービスを登録する。サービスの名称はプロジェクト名称を用いる。なお、def.csv はデフォルト名称であり、Tehaishi 用の cfg.txt、VBScript 用の asp-cfg.txt により、異なる名称のファイルを用いることができる。

初代の MSDE(2001 年)においては、パスワード無しの”sa”アカウントを設定することができた（デフォルト状態）。第二世代の MSSQL7 においては、パスワードを設定することが必須となったが、そのパスワード文字列に関する制約はなかった。第三世代の SQL2012 においては、このパスワードが「強いパスワード」であることを求められるようになった。

人間がこれらのパスワードをキーボード入力する環境であれば、パスワードを紙に記録しておき、必要な時に打鍵すれば良いが、上記のようにプログラムから設定を行い、設定後にサービスやサーバ上のスクリプトで実行するためには、パスワードをプログラム中で固定的に記述するか、外部ファイルにより指定する必要がある。リモートで設定作業を行うためには、外部ファイルはリモートでアクセスできる場所にある必要がある。このことは、セキュリティを保つことが次第に難しくなってきていることを意味する。複雑なパスワードは、コンピュータにとっては入力が簡単であるが、人間にとっては記憶したり間違いなく入力することはより困難である。逆説的であるが、パスワード無しでログインできる条件とし、SQL サーバにはローカルなアクセスのみを許可する方法が最も安全であるように見える。第二世代から、パスワードは、def.csv の中に記述するようになった。これを

利用できるのは、第二世代以後の tehaishi サービスである。

パスワードが不適切であるために処理に失敗した場合、その原因箇所を特定することは手間を要する。このデバッグ作業を容易にするためにアカウントやパスワードを含む豊富なエラーメッセージを出力すると、攻撃者に対して情報を提供する危険性が高まる。

第四世代である三次元データ保管庫においては、専用の検証プログラム（VC-4D、セットアップが終了後は撤去）の一部に組み込んだ機能）により SQL へのプログラムからのログイン条件を検証できるようにし、セットアップが完了した後はパスワードを暗号化したバイナリ形式のファイルの中に埋め込み、プログラムから解読して SQL サーバへのログインに使用している。

リスト 2-6-2 SQL データベース用コネクション文字列

```
"Provider=sqloledb;Data Source=" + su + ";Initial Catalog=" + pDBname;
//これは(local)の場合６階でのみうまく行く。
"Provider=MSDASQL;DRIVER=sql server;SERVER="+su+";DATABASE="+pDBname;//これはＡＳＰと同じ方法。
"Provider=MSDASQL;DRIVER=sql server;SERVER="+su;//データベースを指定せずにつないでみる。
"Provider=MSDASQL;DRIVER={sql server};SERVER="+su+";DATABASE="+pDBname;//これはＡＳＰデータベース作成と同じ方法。
"DRIVER={sql server};SERVER="+su+";DATABASE="+pDBname;//これはＡＳＰと同じ方法。
```

Provider とは、データベースへのアクセスのためにクライアント側でログインに使用する DLL の名称であり、本処理系においては、MSDASQL.dll を使用している。これは記述で省略された場合に使用されるデフォルトの dll である。

Driver とは、ODBC で使用されるデータベースアクセス用のクライアントである。

Server とは、アクセス対象である SQL サーバと、そのシステムがセットアップされているマシン名称を組み合わせたものである。マシン名だけを明記した場合には、SQL サーバとしては既定のサーバが存在すれば、それが指定されたことになる。SQL サーバ名だけを明記した場合、ローカルマシンの上の SQL サーバが指定される。全てが省略された場合には、ローカルマシン上の既定のサーバが指定される。

⑧ SE のための支援機能

設定作業における失敗は、様々な段階で発生する。解決のために、Windows の設定変更、SQL サーバの設定変更、本処理系における定義ファイルの修正で済む場合には、大きな変更は必要ない。この見極めのためには、設定時、修正時、事業終了時に使用する管理者用の VB スクリプト、および Windows 実行形式（サービス等）の設定・デバッグに一時的使用する EXE 形式の診断プログラムが有用である。これらは豊富なエラーメッセージを出す方が親切であるが、現在の状況(Windows2012+SQL2012)に詳細に対応する価値は余りない。しかし、少なくとも以前のシンプルなシステムであった時代に作成したエラーメッセージが、実際とは異なる状況を報告している場合がある点は、修正しておく必要がある。

DB オープンと、SQL コマンド実行のためには、共通のサブルーチンを作成すると共に、エラーに関して詳細にチェックを行い、必要であればログを出力するような処理がデバッグのために有効である。また、このようなサブルーチンを全てのプロジェクトで共通に使用することにより、将来の更なるサーバ間移植の手間を大幅に節約することができる。

しかしながら、便利な機能は、攻撃者のためにも有効なアプローチ手段を提供することにつながる危険性がある。インターネットの利便性よりも悪意ある攻撃のリスクが際立っている 2015 年時点の現状においては、移植に手間がかかる、かなり冗長性のある現行システムのままで当面運用を続けるしかない。

⑨ 簡単なデバッグの方法

デバッグのためのノウハウとして、以下の方法が有効である。

[1]最も基本的な SQL サーバの動作・接続を確認する段階では、単純な処理であるカウンター機能をテストに使用するのが便利である。
[2]プロジェクト毎（地区毎）のサイトを構築する際には、まずコミュニケーションのためのデータベースとそれを構成するテーブル群を、管理機能から作成しておき、同じデータベース名を用いてアクセスカウンタを作成することにより、同一名のデータベース内部にテーブル群を共存させることができる。これらの登録を一挙に実行する機能はまだ存在しない。
[3]複雑な処理を組み合わせた意見投稿コールバックの最後に、Respnse.End の行を挿入し、メッセージをクライアント側に表示させる。
[4]スクリプトが作成するログファイルの内容を調べる（仮登録配下の LogFile ディレクトリ）
[5]W3SVC のログを確認する（コンパイルエラー）

⑩ データベースの保存と移行

第二世代のサーバでは、OS として Windows2003 サーバを使用し、データベースとしては、MSSQL2005 を使用していた。移行先の第三世代のサーバでは、OS として Windows2012 サーバを使用し、データベースとしては、MSSQL2012 を使用した。中間のバージョンを飛ばした移植ではあったが、バックアップファイルに関して、上位互換が保たれていたため、旧版のデータベースを停止することなくバックアップを作成し、これを新たなデータベースにおいて復元することができた。

バックアップに際しては、Microsoft SQL Server Management Studio を使用した。バックアップは、対象となるデータベース毎に行い、選択・右クリックで現れるポップアップから、タスク→バックアップで、操作画面を開く。次に、以下の設定を行う。

「全般」ページでの設定
・バックアップの種類：[完全]を選択
・バックアップコンポーネント：[データベース]にチェック
・バックアップ先：[ディスク]にチェックしバックアップを格納するディレクトリを指定
「オプション」ページでの設定
・メディアに上書きします
[既存のメディアセットにバックアップする]をチェック
→[既存の全てのバックアップセットを上書きする]をチェック
（追加する、という設定では、復元段階で障害が生じる）
・信頼性　[完了時にバックアップを検証する]にチェック
[メディアに書き込む前にチェックサムを行う]にチェック

以上の設定を行った上で、OK ボタンによりバックアップを行う。バックアップ処理は数秒で完了する。

記憶媒体もしくは FFFTP 等を用いて、バックアップを新たな環境にコピーする。

復元は、予めデータベースが作成してある状態から出発し、同様に Microsoft SQL Server Management Studio を使用して作業を行った。既に作成してあるデータベースを選択・右クリックし、タスク→復元→データベースで、操作画面を開く。

「全般」ページで、
・復元するバックアップセットの復元元ファイルと場所：デバイスからとして、ファイルを選択する。
「オプション」ページで、
・復元オプション：既存のデータベースを上書きする（WITH　REPLACE）
・復元先：（新たなデータベースの）mdf ファイルと、ldf ファイルをフルパスで記述

以上の設定の上で、OK を押すと、現在のデータベースの上に、復元が行われる。復元は、バックアップよりも時間を要する。

このとき、クエリ等が開いていると、「アクセス権が取得できませんでした」というエラー終了となる。また、上記の「上書き」が選択されていない場合には、「ログの末尾が　アックアップされませんでした。」というエラー終了となる。

バックアップファイルの名称は随意であるが、.bak 等の拡張子が用いられている。

⑪ まとめ

仮想化を伴う、異なる OS への移築による保存継承の手順は、W2003→W2012 の移行を例とすると、以下のような流れとなる。

[1] DVD-ROM や、USB メモリ等のメディアから新たな OS を導入し、その途中でコマンドラインから VHD を作成してそこに新たな OS を導入する。
[2] 新たな OS 上で、IIS を導入する。
[3] 新たな OS 上で、SQL エンジンを導入する。
[4] 新たな IIS 上で、既存のプロジェクト別の仮想ディレクトリを作成する。
[5] 新たな IIS 上で、ASP の実行環境を設定する（エラー処理等）。
[6] 古いデータベースをバックアップし、新たな SQL エンジン上に復元する。
[7] 新たな IIS 上で、FTP のアクセス環境を作成する。
[8] 古い実行環境を VHD に変換し、ファイルとしてアーカイブする。

W2003+SQL7 から W2012＋SQL2012 への移植にあたって、設定変更（条件緩和）により旧システムとの互換性を実現することができた項目は、以下の通りである。

[1] 32 ビットのサービスを許容するように IIS を設定する
[2] 上位ディレクトリへの参照を許容するように設定する
親ディレクトリの参照方法として、”..”を使用することは初期状態ではできなくなった。使用するためには IIS での設定を行い、再起動する手順が求められる。これにより、インクルードファイルがアクセスできないことによるコンパイルエラーを防ぐことができ、ま

た移植に伴い必要となる設定ファイル def.csv の修正箇所を少なくすることができる。
[3] VBScript のコンパイルエラーをクライアント側に表示する方法は、可能と解説されているが、いずれの方法も失敗した。サーバ側に開発・デバッグ環境をセットアップしておき、サーバ上で直接見る以外の方法はなさそうである。

コンパイルが成功した後には、スクリプトでクライアント側に返すメッセージとして、あるいはログファイルとして、デバッグのために有用な情報を得ることができる。

結果的に、アプリケーション側で以下のようなプログラムないしアプリケーション内部の設定ファイルの修正を行った。

[1] asp-cfg.txt を修正し、def.csv が存在する場所をスクリプトに正しく伝える（２か所）
[2] def.csv の内容を修正する（１か所）
[3]SQL サーバのアカウントとパスワードを記述する tdb_user.asp を修正する（３か所）
SQL パスワードの修正箇所は、VB スクリプト用３か所、サービス用１か所である。 SQL サーバへのパスワードを修正する。パスワードは必ず設定しなければ動作しない。しかも、強いパスワードである必要がある。MSDE ではパスワード無しで動作した。MSSQL7 では、弱いパスワードでも動作した。
[4]データベースを新規に作成する VB スクリプト（dbmake、SQL 文字列固定記述）において、MDF ファイルのサイズを１MB から５MB に増量する。修正前の失敗に関して、従来のシステムでは、「データベースは既にあります」というメッセージを発するため、設定者は混乱する。
[5]投稿用のコールバック関数 VB スクリプトを修正する。
コメントアウトされている on Error Resume Next を有効にする（２か所）

２－７．ＶＣ－４Ｄ　三次元データ保管庫

（１）概要

仮想コンバータを用いたデータベース入出力プログラム（以下、本システム）は、サーブレットエンジン（Apache Tomcat）上で動作する Web アプリケーションである。

本システムでは、主に以下の３機能を有する。

① アップロード受付機能

ユーザが作成した形式記述ファイル（メタファイル）と形状記述ファイル（データファイル）を受け付け、サーバ上の所定のフォルダ下に保存する。

保存した各ファイルを仮想コンバータへ受け渡し、３次元データをデータベース上に展開する。

② ダウンロード受付機能

ユーザが作成した形式記述ファイル（メタファイル）と登録済データ（選択式）を受け付け、サーバ上の所定のフォルダ下に保存する。

保存したファイルと選択した登録済データを仮想コンバータへ受け渡し、サーバ

上に形状記述ファイルを生成する。

③ 履歴参照機能

ユーザが行ったアップロード／ダウンロード要求を、一覧／詳細表示する機能。

各要求内容の進捗状況や、処理結果（合否）内容が参照できる。

ダウンロード要求の場合は、生成された形状記述ファイルのダウンロードが行える。

（２）動作環境

①動作環境

表 2-7-1 VC-4D の動作環境

項目	内容
サーバ OS	Windows 7, Windows XP
クライアント Web ブラウザ	IE 8 以降、Mozilla Firefox 18 以降

※ 本システムの動作試験を行った環境

②利用アプリケーション一覧

表 2-7-2 VC-4D の利用アプリケーション

項目	内容
サーブレットエンジン	Apache Tomcat 7.0.35
Java SDK	JDK 7
HTTP サーバ	Apache HTTP Server 2.2
データベース	Microsoft SQL Server 2008 R2 SP1 Express Edition （本システムの動作試験で使用）

（３）システムのインストール

①Java SDK

jdk-7-windows-i586.exe（64 ビット OS の場合は、jdk-7-windows-x64.exe）より、Java SDK のインストールを行う。

以下は、32 ビット OS 上でインストールを行う場合の例である。

1) jdk-7-windows-i586.exe をダブルクリックする。

2) インストールウィザードに従い、デフォルトで指定される内容で各画面を進める。

3) インストール終了。

②Apache Tomcat

1)インストール

apache-tomcat-7.0.35.exe より、Apache Tomcat のインストールを行う。

リスト 2-7-1 Apache Tomcat のインストール手順

(1) apache-tomcat-7.0.35.exe をダブルクリックし、インストールウィザードを表示させる。
(2) 「License Agreement」画面では、”I Agree” ボタンを選択する。
(3) 「Choose Components」画面では、デフォルトの内容で”Next” ボタンを選択する。

(4) 「Configuration」画面では、デフォルトの内容で”Next” ボタンを選択する。
(5) 「Java Virtual Machine」画面では、デフォルトの内容で”Next” ボタンを選択する。
(6) 「Choose Install Location」画面では、デフォルトで指定されるインストールフォルダ以外を選択する。
【例】C:¥Apps¥Apache Software Foundation¥Tomcat 7.0
【理由】
Windows Vista 以上の OS において、User Account Control（UAC）を有効化している場合は、OS 管理下におかれているフォルダ（例　C:¥Program Folder）下にインストールを行った場合、Web アプリケーションが正常に動作しない可能性があるため。
(7) ファイルが展開され、インストール終了。
(8) Windows ファイアーウォールにおいて、ポート番号 8080（HTTP）と 8009（AJP）でのアクセス許可を設定する。　※ 外部 Web サーバと連携する場合のみ

2) 起動／停止

Windows スタートメニューより、「Apache Tomcat 7.0 Tomcat7」→「Monitor Tomcat」を指定しダイアログを表示させる。

ダイアログ上部の「General」タブを指定し、”Service Status:” 下部にある “Start” ボタンで起動、“Stop” ボタンで停止を行う。

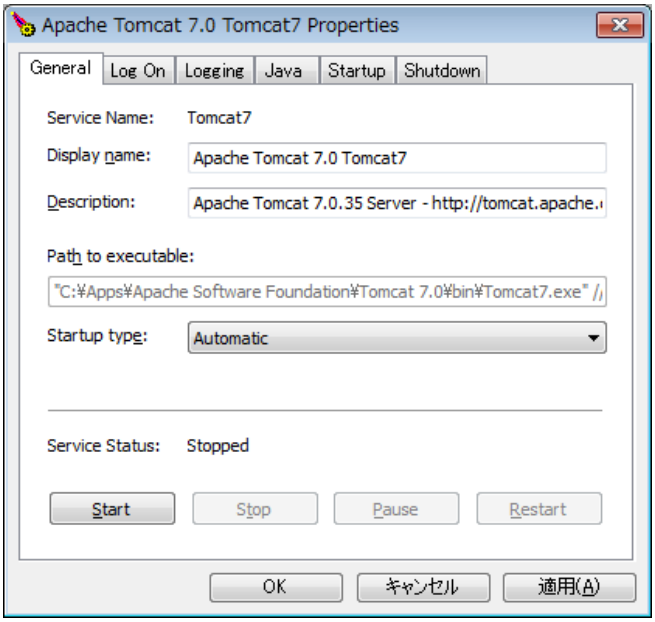


図 2-7-1 Apache Tomcat 起動／停止ダイアログ

③ Web アプリケーション（本システム）

1) インストール

vc4d.war を、サーブレットエンジンのアプリケーションフォルダ下にコピーし、リスト 2-7 の操作を行う。

リスト 2-7-2 WEB アプリケーションのインストール手順

(1) Apache Tomcat を停止する。
(2) Apache Tomcat インストールフォルダ中の、webapps フォルダ（アプリケーションフォルダ）を開く。
(3) 上記フォルダへ、vc4d.war ファイルをコピーする。
(4) Apache Tomcat を起動する。
(5) 起動後、アプリケーションフォルダ内に vc4d フォルダが生成されていることを確認する。

2) 設定変更

Web アプリケーションの設定変更を行うには、共通設定プロパティファイル（vc4d.properties）を編集する。

基本的に設定変更が必要な項目は、以下の通り。

- アップロード／ダウンロード要求で受け付けたファイルを保存するフォルダ。
- 仮想コンバータプログラムのパス指定。
- データベースの接続情報。

共通設定プロパティファイルは、

${TOMCAT_HOME}/webapps/vc4d/WEB-INF/classes

フォルダ下に存在する。${TOMCAT_HOME} はサーブレットエンジン インストールフォルダを示す。

共通設定プロパティファイルにおいて設定可能な項目の詳細は、(6)③1) を参照。

1. Apache Tomcat を停止する。
2. ${TOMCAT_HOME}/webapps/vc4d/WEB-INF/classes/vc4d.properties を編集し、保存する。
3. Apache Tomcat を起動する。

④データベース設定

本システムがデータベースへアクセスするために、Microsoft SQL Server 2008 R2 SP1 Express Edition の設定を変更する。

尚、設定変更は付属の SQL Server 2008 R2 SP1 Management Studio Express を利用する。

1) サーバ認証方法の変更

Windows スタートメニューより、「Microsoft SQL Server 2008 R2」→「SQL Server Manager Studio」を指定し起動する。

画面左部の「オブジェクト エクスプローラ」に表示されるインスタンス名称を右クリックし、プロパティを表示させる。

プロパティダイアログ中の「セキュリティ」ページを選択し、「サーバ認証」を “SQL Server 認証モードと Windows 認証モード” に変更する。

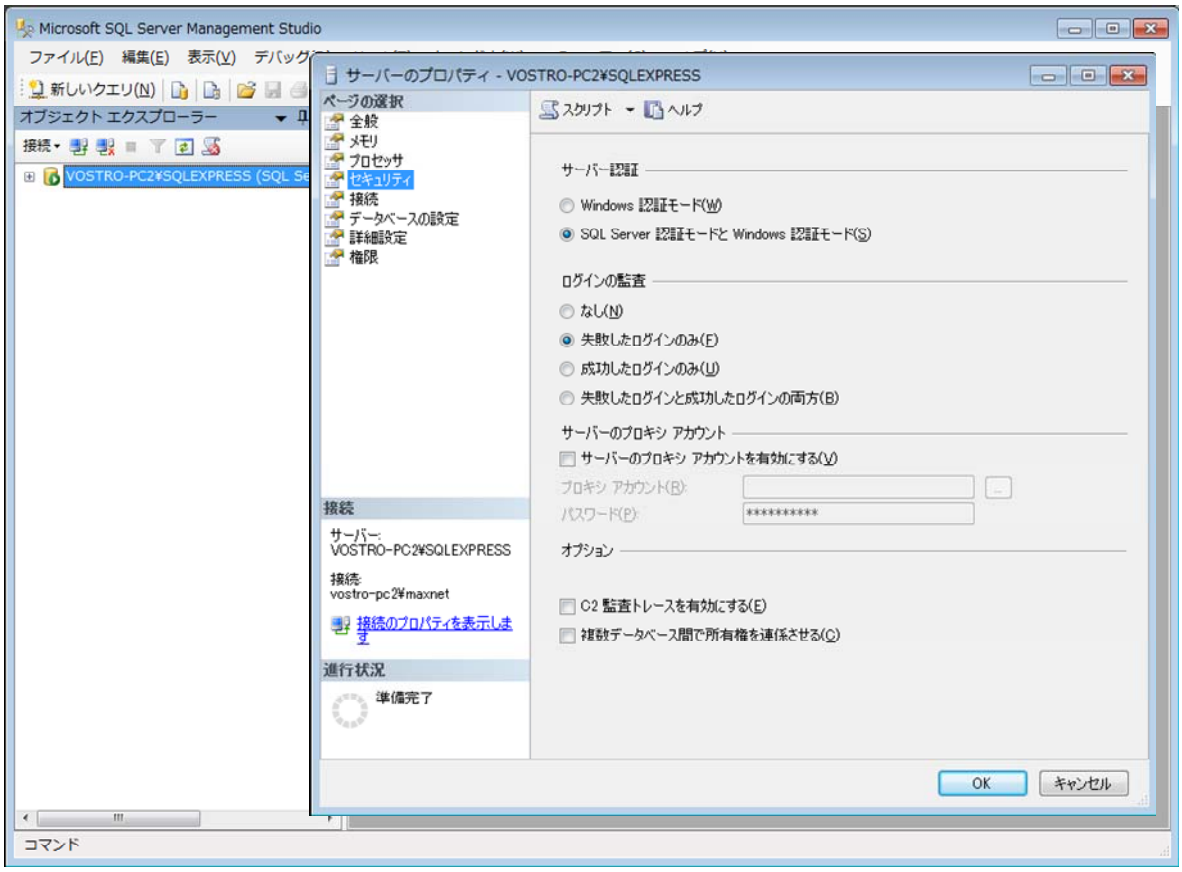


図 2-7-2 SQL サーバ認証方法変更

次に、データベースへのログインに使用する sa ユーザの有効化を行う。

画面左部の「オブジェクト エクスプローラ」より、「セキュリティ」→「ログイン」下に表示される sa ユーザ名称を右クリックし、プロパティを表示させる。

プロパティダイアログ中の「状態」ページを選択し、「データベースエンジンに接続する権限」項目を“許可”、「ログイン」項目を“有効”に変更する。

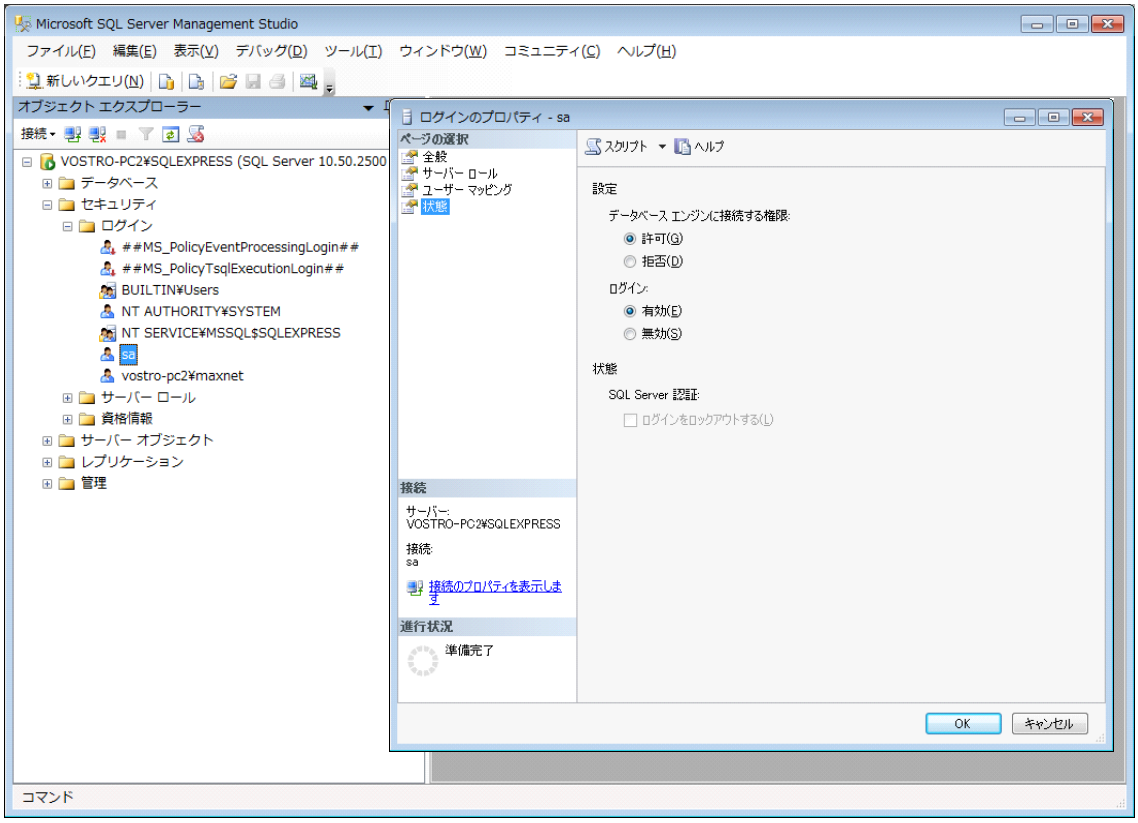


図 2-7-3 SQL サーバ sa ユーザの有効化

2) TCP／IP の変更

Windows スタートメニューより、「Microsoft SQL Server 2008 R2」→「構成ツール」→「SQL Server 構成マネージャ」を指定し起動する。

画面左部の「SQL Server ネットワーク構成」より、使用するインスタンスのプロトコルを選択する。

画面右部に表示される「TCP/IP」項目を右クリックし、「プロパティ」を選択する。

プロパティ画面上部の「IP アドレス」を選択し、SQL Server で接続を許可する IP アドレスを以下の内容で変更する。（localhost で接続する場合は、IP アドレス ”127.0.0.1” または ”::1” を変更）

表 2-7-3 SQL サーバの TCP/IP 設定(local host)

項目	設定内容
TCP ポート	1433
TCP 動的ポート	（空白）
アクティブ	はい
有効	はい

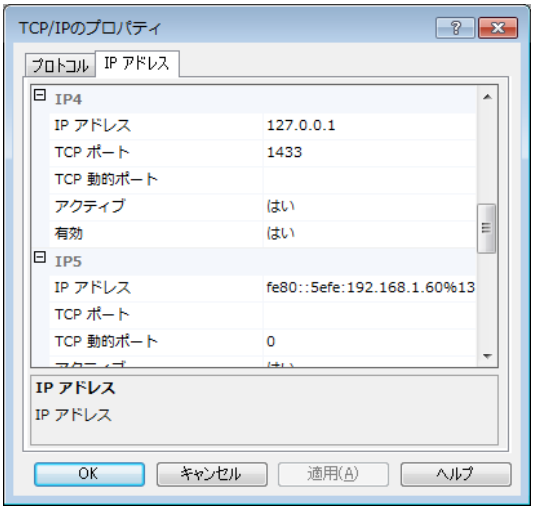


図 2-7-4 SQL サーバ TCP/IP プロパティ変更(local host)

続いて、プロパティ画面最下部にある“IPAll”の項目を、以下の内容で変更する。

表 2-7-4 SQL サーバの TCP/IP 設定(IPAll)

項目	設定内容
TCP ポート	1433
TCP 動的ポート	（空白）

図 2-7-5 SQL サーバ TCP/IP プロパティ変更（IPAll）

最後に、画面右部に表示される「TCP/IP」項目を右クリックし、「有効化」を選択する。

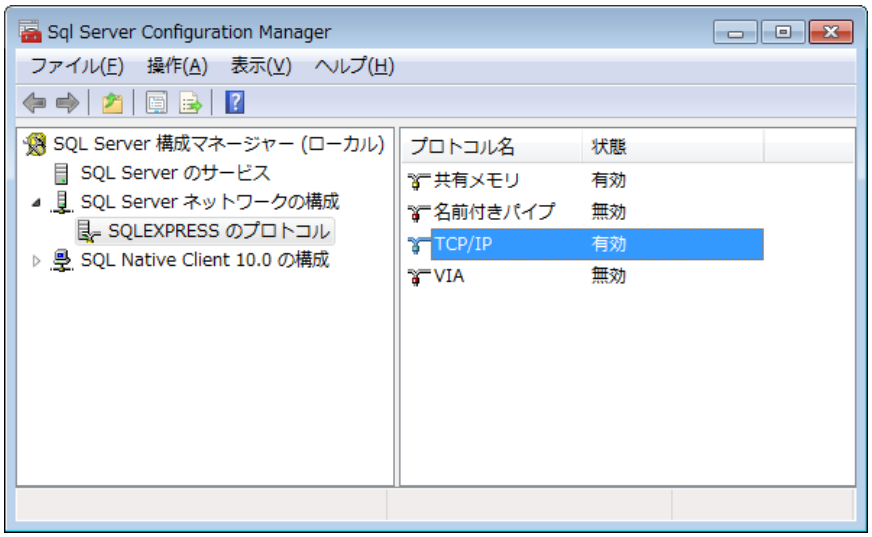


図 2-7-6 SQL サーバ TCP/IP 有効化

3) システム用データベース／テーブルの作成

SQL Server 上に本システムで利用するデータベースおよびテーブルを作成する。

データベース名称は vc4d とする。

利用するテーブルは Master と JobTask の 2 つで、その用途を表 2-9 に示す。

表2-7-5 VC-4Dシステム用テーブルの用途

テーブル名	用途
Master	仮想コンバータにおいて3次元データを展開したデータベース名を管理。
JobTask	仮想コンバータの実行スケジュールを管理。

Master テーブルの各カラム内容は、表 2-10 の通り。

表 2-7-6 VC-4D システム用 Master テーブルの列構成

項目名	項目 ID	属性	必須	説明
データベース ID	Id	Int	○	自動採番されるユニーク ID。
データベース名	Name	Varchar(256)		3次元データが展開されたデータベース名称。

JobTask テーブルの各カラム内容は、下記の通り。

表 2-7-7 VC-4D システム用 JobTask テーブルの列構成

項目名	項目 ID	属性	必須	説明
受付 ID	Id	Varchar(64)	○	アップロード／ダウンロード要求受付時に発行するユニーク ID。
セッション ID	Sid	Varchar(64)	○	ユーザの Web ブラウザに Cookie データとして登録されたセッション ID。
処理種別	JobType	Smallint	○	アップロード要求＝1、ダウンロード要

				求＝2。
処理状態	Status	Smallint	○	未処理＝0、処理中＝1、処理済＝2
処理結果	Result	Bit	○	仮想コンバータでの処理に成功した場合＝1
処理結果コード	ResultCode	Int	○	仮想コンバータからの返却値
処理メッセージ	Message	Varchar(1024)		エラーメッセージなど。
お名前	Name	Varchar(256)		ブラウザで入力した「お名前」
保存フォルダ	Folder	Varchar(512)	○	ファイルが保存されているフォルダ名
メタファイル名	MetaFile	Varchar(256)		サーバ内で保存される形式定義ファイル名
メタファイル名	MetaOrgFile	Varchar(256)		ブラウザで指定した「形式定義ファイル」のファイル名
データファイル名	DataFile	Varchar(256)		サーバ内で保存される形状記述ファイル名
データファイル名	DataOrgFile	Varchar(256)		ブラウザで指定した「形状記述ファイル」のファイル名
データベース名	DbName	Varchar(256)		(アップロード)仮想コンバータで作成するデータベース名称 (ダウンロード)ブラウザで選択した「登録済みデータ」
登録日時	RegistDate	DateTime	○	処理を受け付けた日時。
開始日時	StartDate	DateTime		仮想コンバータを実行した日時。
終了日時	FinishDate	DateTime		仮想コンバータの処理が終了した日時。

データベースは、SQL Server Manager Studio より作成する。

画面左部の「オブジェクト エクスプローラ」に表示されるインスタンス名称を選択し、下段に現れる「データベース」を右クリック→「新しいデータベース」を選択する。

データベース作成用ダイアログの画面右上部にある「データベース名称」へ ”vc4d” と入力し、OK ボタンを選択する。

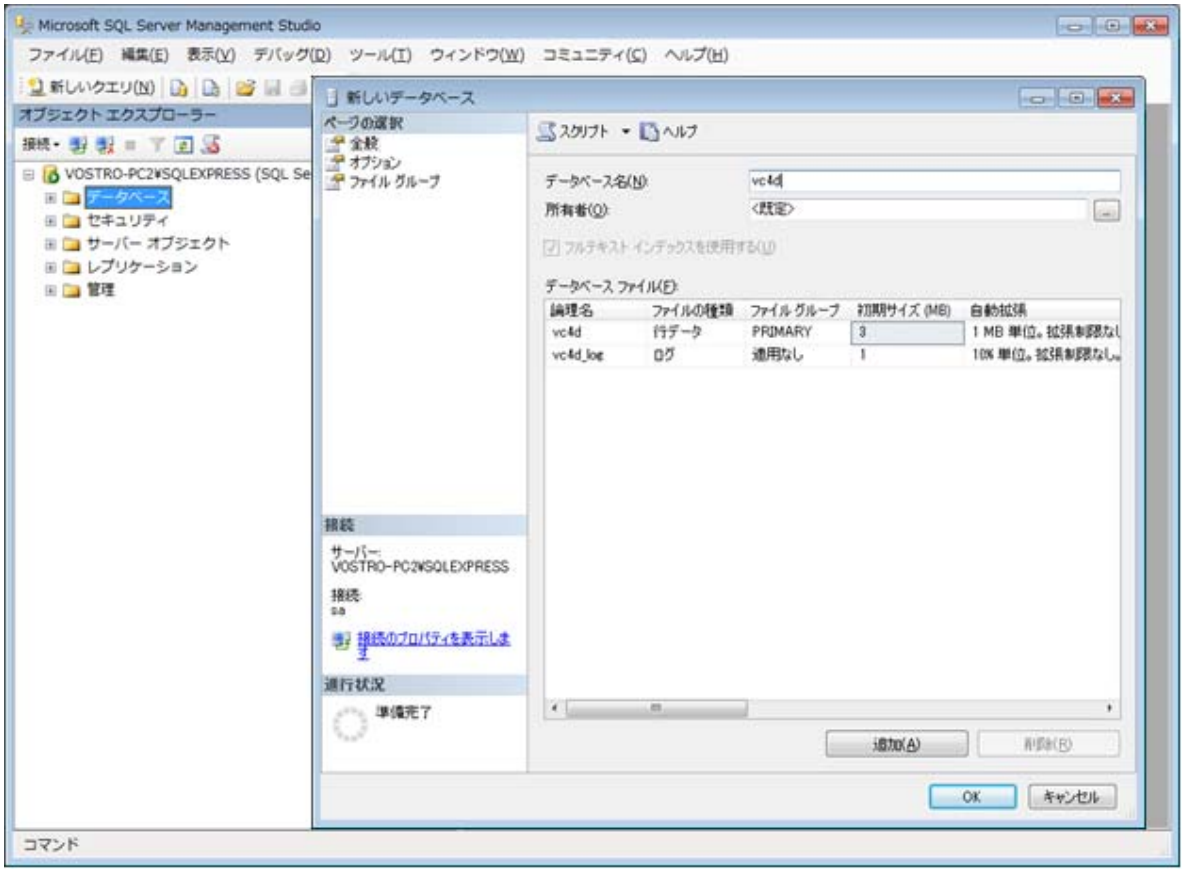


図 2-7-7 SQL サーバ　新規データベース作成

各テーブルは、SQL Server Manager Studio より、以下のクエリを実行して作成する。

リスト 2-7-3 テーブル作成用クエリ

```
USE [vc4d]
GO

SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
SET ANSI_PADDING ON
GO

CREATE TABLE [dbo].[Master](
	[Id] [int] IDENTITY(1,1) NOT NULL,
	[Name] [varchar](256) NULL
) ON [PRIMARY]
GO

CREATE TABLE [dbo].[JobTask](
	[Id] [varchar](64) NOT NULL,
	[Sid] [varchar](64) NULL,
	[JobType] [smallint] NOT NULL,
	[Status] [smallint] NOT NULL,
	[Result] [bit] NOT NULL,
	[ResultCode] [int] NOT NULL,
	[Message] [varchar](1024) NULL,
	[Name] [varchar](256) NULL,
```

```
	[Folder] [varchar](512) NOT NULL,
	[MetaFile] [varchar](256) NULL,
	[MetaOrgFile] [varchar](256) NULL,
	[DataFile] [varchar](256) NULL,
	[DataOrgFile] [varchar](256) NULL,
	[DbName] [varchar](256) NULL,
	[RegistDate] [datetime] NOT NULL,
	[StartDate] [datetime] NULL,
	[FinishDate] [datetime] NULL
) ON [PRIMARY]
GO

SET ANSI_PADDING OFF
GO
```

⑤ Web サーバ

1) インストール

httpd-2.2.22-win32-x86-no_ssl.msi より、Web サーバ（Apache HTTP Server）のインストールを行う。

尚、インストールを行う環境上に別の Web サーバ（IIS 等）が導入されている場合はポート番号が衝突するため、サービスを停止する必要がある。

リスト 2-7-4 Web サーバ Apache のインストール手順

(1) httpd-2.2.22-win32-x86-no_ssl.msi をダブルクリックし、インストールウィザードを表示させる。
(2) 「License Agreement」画面では、"I Agree" ボタンを選択する。
(3) 「Reed This First」画面では、"Next" ボタンを選択する。
(4) 「Server Information」画面では、各項目に適切な値を入力し"Next" ボタンを選択する。
(5) 「Setup Type」画面では、デフォルトの内容で"Next" ボタンを選択する。
(6) 「Destination Folder」画面では、デフォルトで指定されるインストールフォルダ以外を選択する。
【例】C:¥Apps¥Apache Software Foundation¥Apache2.2
【理由】Windows Vista 以上の OS において、User Account Control（UAC）を有効化している場合は、OS 管理下におかれているフォルダ（例　C:¥Program Folder）下にインストールを行った場合、Web サーバの設定が反映されない可能性があるため。
(7) ファイルが展開され、インストール終了。
(8) Windows ファイアーウォールにおいて、ポート番号 80（HTTP）でのアクセス許可を設定する。

2) 設定変更

Windows スタートメニューより、「Apache HTTP Server 2.2」→「Configure Apache Server」→「Edit the Apache httpd.conf Configuration File」を選択し、テキストエディタを表示させる。

テキストエディタで行末までスクロールし、以下の内容を追加後に保存・終了する。

リスト 2-7-5 Apache 設定ファイル末尾への追加項目

```
LoadModule proxy_module modules/mod_proxy.so
LoadModule proxy_ajp_module modules/mod_proxy_ajp.so

<Location /vc4d>
  ProxyPass ajp://localhost:8009/vc4d
</Location>
```

3) 起動／停止

Windows スタートメニューより、「Apache HTTP Server 2.2」→「Control Apache Server」を選択し、下に表示される「Start」「Stop」メニューより起動／停止を行う。

（4）システムのアンインストール

① Web アプリケーション（本システム）

以下に、本システムのアンインストール方法を記す。

リスト 2-7-6 本システムのアンインストール手順

(1) Apache Tomcat を停止する。
(2) Apache Tomcat インストールフォルダ中の、webapps フォルダ（アプリケーションフォルダ）を開く。
(3) 上記フォルダ中の、vc4d.war ファイルと vc4d フォルダを削除する。
(4) アップロードにより保存されたファイルは、共通設定プロパティファイルに指定したフォルダ下に存在する。
これらのファイルはアンインストールの対象とならないため、手動で削除を行う。
（バックアップが必要な場合は、任意の圧縮ツールを使用して行うこと）

②その他のアプリケーション

Java SDK／Apache HTTP Server／Apache Tomcat は、Windows コントロールパネルの「プログラムと機能」より、アンインストールを行う。

本システムで利用するデータベース（vc4d）や仮想コンバータより登録された 3 次元データベースは、SQL Server Manager Studio より手動で削除を行う。

画面左部の「オブジェクト エクスプローラ」に表示されるデータベース名称を選択し、右クリック→「削除」を指定する。

データベース削除用ダイアログにおいて、OK ボタンを選択する。

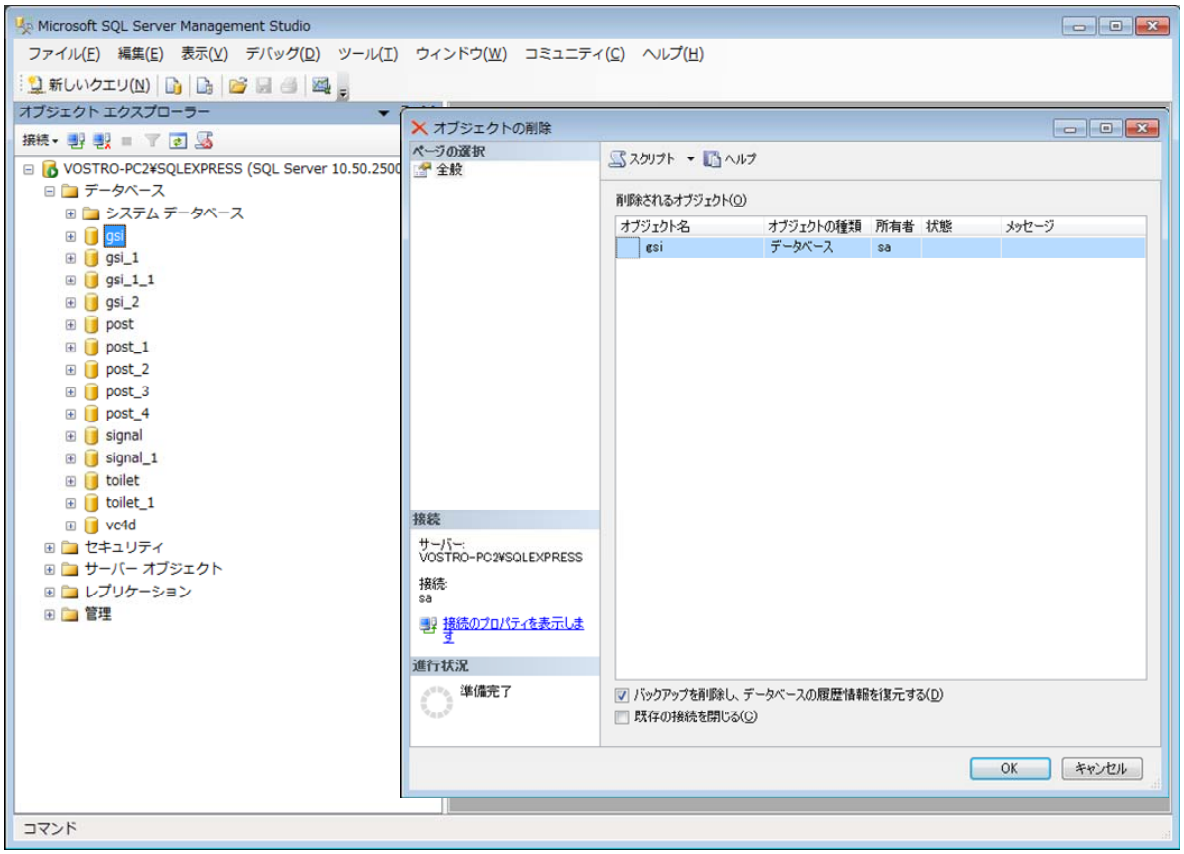


図 2-7-8 SQL サーバデータベース削除

（５）システム構成

①Web アプリケーション ファイル構成

１）全体構成

サーブレットエンジンのインストールフォルダ下に展開される、本システムのフォルダ／ファイル構成。

```
${TOMCAT_HOME}                                          Apache Tomcat インストールフォルダ
  ├ logs/                                               ログ出力フォルダ
  └ webapps/                                            Web アプリケーションフォルダ
     └ vc4d/                                            本システム アプリケーションフォルダ
        ├ META-INF/
        ├ resources/
        │  └ css/
        │     └ style.css                               スタイルシートファイル
        │
        └ WEB-INF/
           ├ classes/
           │  ├ vc4d/                                   クラスファイル格納フォルダ
           │  ├ vc4d.properties                         共通設定プロパティファイル
           │  ├ job-messages.properties                 仮想コンバータ用メッセージファイル
           │  ├ applicationContext.xml
           │  ├ dataAccessContext.xml
           │  ├ webContext.xml
           │  ├ messages-spring_jp.propertiesl
           │  └ logback.xml                             ログ出力定義 XML
           │
           ├ lib/
           ├ view/                                      JSP ファイル格納フォルダ
           │  ├ download-index.jsp
           │  ├ parts-footer.jsp
           │  ├ parts-header.jsp
           │  ├ parts-menu.jsp
           │  ├ parts-status-detail.jsp
           │  ├ status-detail.jsp
           │  ├ status-index.jsp
           │  ├ status-search.jsp
           │  ├ upload-index.jsp
           │  ├ http-404.jsp
           │  └ http-500.jsp
           │
           └ web.xml                                    Web アプリケーション配備記述子 XML


${仮想コンバータ用データフォルダ}
  ├ exe/                                                仮想コンバータコンソール格納フォルダ
  ├ data/                                               3 次元データベースファイル格納フォルダ
  └ file/                                               アップロードファイル格納フォルダ
```

図 2-7-9 本システムのフォルダ／ファイル構成

2）Web アプリケーション リソースファイル構成

本システムで使用する、各リソースファイルの構成。

各ファイルの内容を変更することで、システム動作の変更が行える。

基本的に内容変更が必要なファイルは、“vc4d.properties”と“job-messages.properties”の２ファイルのみ。

尚、Web アプリケーションをアンインストールした場合、これらのファイルも削除されるため、内容変更を行ったファイルは任意の場所にバックアップを行うこと。

```
${TOMCAT_HOME}/webapps/vc4d/WEB-INF/classes/
  ├ vc4d.properties                 共通設定プロパティファイル
  ├ job-messages.properties         仮想コンバータ用メッセージファイル
```

```
  ├ applicationContext.xml                    Springframework 定義 XML
  ├ dataAccessContext.xml                     （同上）
  ├ webContext.xml                （同上）
  ├ messages-spring_jp.propertiesl            メッセージプロパティファイル
  └ logback.xml                               ログ出力定義 XML
```

図 2-7-10 本システムのリソース構成

3）Web アプリケーション HTML 画面構成

Web ブラウザで表示される、各 HTML 画面の構成。

テキストエディタ等で編集することで、表示する HTML の内容が変更される。

尚、Web アプリケーションをアンインストールした場合、これらのファイルも削除されるため、内容変更を行ったファイルは任意の場所にバックアップを行うこと。

```
${TOMCAT_HOME}/webapps/vc4d/
  ├ WEB-INF/
  |   └ views/
  |       ├ download-index.jsp          ダウンロード受付用 JSP ファイル
  |       ├ parts-footer.jsp            （共通）フッタ用 JSP ファイル
  |       ├ parts-header.jsp            （共通）ヘッダ用 JSP ファイル
  |       ├ parts-menu.jsp                      （共通）メニュー用 JSP ファイル
  |       ├ parts-status-detail.jsp             （共通）履歴詳細用 JSP ファイル
  |       ├ status-detail.jsp           履歴一覧用 JSP ファイル
  |       ├ status-index.jsp            履歴詳細用 JSP ファイル
  |       ├ status-search.jsp           履歴検索用 JSP ファイル
  |       ├ upload-index.jsp            アップロード受付用 JSP ファイル
  |       ├ http-404.jsp                        HTTP ステータスコード 404 用 JSP ファイル
  |       └ http-500.jsp                        HTTP ステータスコード 500 用 JSP ファイル
  |
  └ resources/
      └ css/
          └ style.css                  スタイルシートファイル
```

図 2-7-11 HTML 画面構成

4）ログファイル構成

サーブレットエンジンおよび本システムで出力される、ログファイルの構成。

日単位にファイル名の“YYYY-MM-DD”が変更されて出力される。

```
${TOMCAT_HOME}/logs/
  ├ vc4d.YYYY-MM-DD.log                       本システムより出力されるログ。
  ├ catalina.YYYY-MM-DD.log          Apache Tomcat 全体のログ。
  ├ localhost.YYYY-MM-DD.log         本システムで発生した例外が出力されるログ。
  ├ localhost_access_log.YYYY-MM-DD.txt 本システムへのアクセスログ。
  └ tomcat7-stdout.YYYY-MM-DD.log             Apache Tomcat 標準エラー出力のログ
```

図 2-7-12 ログファイル構成

システムエラー等が発生した場合は、以下の順序で各ファイルの内容を調査する

① vc4d.YYYY-MM-DD.log

② localhost.YYYY-MM-DD.log

③ catalina.YYYY-MM-DD.log

④ tomcat7-stdout.YYYY-MM-DD.log

５）仮想コンバータ用データフォルダ構成（例）

仮想コンバータ用データフォルダの構成は、本システムの共通設定プロパティファイルで定義する。

「仮想コンバータコンソールアプリケーションパス」「３次元データベースファイル格納フォルダ」「ダウンロード要求ファイル格納フォルダ」「アップロード要求ファイル格納フォルダ」は、任意のパスが指定可能。

システムにアップロードされたファイルは、「ダウンロード要求ファイル格納フォルダ」「アップロード要求ファイル格納フォルダ」の配下に、「年／月／日／受付 ID」フォルダが作成されて保存される。

図 2-7-13 に、データフォルダの構成例を示す。

```
${仮想コンバータ用データフォルダ}
  ├ exe/                                  仮想コンバータコンソール格納フォルダ
  │   └ vc4d.exe                          仮想コンバータコンソールアプリケーション
  │
  ├ data/                                 ３次元データベースファイル格納フォルダ
  │   ├ gis.ldf                           ３次元データ展開データベースファイル
  │   └ gsi.mdf                                 （ファイル名は例）
  │
  └ file/
      ├ download/                         ダウンロード要求ファイル格納フォルダ
      │   └ YYYY/
      │       └ MM/
      │           └ DD/
      │               └ 受付 ID/
      └ upload/                           アップロード要求ファイル格納フォルダ
          └ YYYY/
              └ MM/
                  └ DD/
                      └ 受付 ID/
```

図 2-7-13 データフォルダ構成例

②ソースコード ファイル構成

vc4d-src.zip を、任意のフォルダ下に展開する。

展開後の各フォルダ構成は、下記の通り。

```
vc4d/                                     ソースコードフォルダ
  ├ build.sh                              ビルド用スクリプト
  ├ pom.xml                               Maven プロジェクト記述ファイル
  ├ db/                                   テスト用 DB 定義ファイル
  ├ dummy/                                テスト用 仮想コンバータプログラム
  └ src/
      ├ main/
      │   ├ java/                         Java ソースコード
      │   ├ resources/                    リソースファイル
      │   └ webapps/                      Web 画面ファイル
      │
      └ test/                             ユニットテスト用モジュール SET ANSI_PADDING OFF
```

図 2-7-14 ソースコードファイル構成

１）Java ソースコードファイル一覧

Java ソースコードファイルの構成。

```
vc4d/src/main/java/
  └vc4d/
    ├ core/
    │   ├ dao/                                  Data Access Object パッケージ
    │   │   ├ GenericDao.java
    │   │   ├ JobTaskDao.java
    │   │   ├ ShapeDao.java
    │   │   └ jpa/
    │   │       ├ GenericDaoImpl.java
    │   │       ├ JobTaskDaoImpl.java
    │   │       └ ShapeDaoImpl.java
    │   │
    │   ├ entity/                               データエンティティパッケージ
    │   │   ├ JobTask.java
    │   │   └ Shape.java
    │   │
    │   └ service/                          各サービスモジュールパッケージ
    │       ├ DataService.java
    │       ├ FileService.java
    │       └ impl/
    │           ├ DataServiceImpl.java
    │           └ FileServiceImpl.java
    │
    ├ job/                                  バックグラウンドジョブパッケージ
    │   └ CommandExecutor.java
    │
    └── web/
        ├ controller/                           Web アプリケーションパッケージ
        │   ├ AbstractCookieController.java
        │   ├ DownloadController.java
        │   ├ IndexController.java
        │   ├ StatusController.java
        │   └ UploadController.java
        │
        ├ form/                             Web フォームパッケージ
        │   ├ DownloadForm.java
        │   └ UploadForm.java
        │
        └ listener/                             Web リスナーパッケージ
            └ LocalContextCleanupListener.java
```

図 2-7-15 JAVA ソースコードファイル構成

2）リソースファイル一覧

各リソースファイルの構成。

```
vc4d/src/main/resources/
  ├ vc4d.properties                     共通プロパティファイル
  ├ job-message.properties                      仮想コンバータ用メッセージファイル
  ├ applicationContext.xml                      Springframework 定義 XML
  ├ dataAccessContext.xml                       （同上）
  ├ webContext.xml                  （同上）
  ├ messages-spring_jp.properties               メッセージプロパティファイル
  └ logback.xml                                 ログ出力定義 XML
```

図 2-7-16 リソースファイル構成

3）Web 画面ファイル一覧

Web ブラウザで表示される、各 HTML 画面の構成。

```
vc4d/src/main/webapp/
  ├ WEB-INF/
  │   ├ views/
  │   │   ├ download-index.jsp          ダウンロード受付用 JSP ファイル
  │   │   ├ parts-footer.jsp            （共通）フッタ用 JSP ファイル
  │   │   ├ parts-header.jsp            （共通）ヘッダ用 JSP ファイル
  │   │   ├ parts-menu.jsp                    （共通）メニュー用 JSP ファイル
  │   │   ├ parts-status-detail.jsp           （共通）履歴詳細用 JSP ファイル
  │   │   ├ status-detail.jsp           履歴一覧用 JSP ファイル
  │   │   ├ status-index.jsp            履歴詳細用 JSP ファイル
  │   │   ├ status-search.jsp           履歴検索用 JSP ファイル
  │   │   ├ upload-index.jsp            アップロード受付用 JSP ファイル
  │   │   ├ http-404.jsp                      HTTP ステータスコード 404 用 JSP ファイル
  │   │   └ http-500.jsp                      HTTP ステータスコード 500 用 JSP ファイル
  │   │
  │   └ web.xml                               Web アプリケーション配備記述子 XML
  │
  └ resources/
      └ css/
          └ style.css                   スタイルシートファイル
```

図 2-7-17 Web 画面ファイル構成

4）ソースコードのビルド

ソースコードのビルドは、Apache Maven ビルドツールを用いて行う。

以下、ビルドの手順を説明する。

リスト 2-7-7 ソースコードのビルド手順

```
(1) apache-maven-3.0.4-bin.zip を、任意のフォルダ下に展開する。
(2) 展開後、コマンドプロンプトを開き、ソースコードフォルダへ移動する。
(3) Java SDK をインストールしたフォルダを、環境変数 JAVA_HOME に設定する。
(4) mvn.bat を package オプションを付与して実行し、ソースコードのビルドを行う。
      mvn.bat は Apache Maven を展開したフォルダ下の、bin フォルダに存在する。
(5) ビルドが実行され、ソースコードフォルダ下の target フォルダに vc4d.war が生成される。
```

下の例は、C:¥vc4d-build フォルダ下に Apache Maven とソースコードを展開後、ビルドを行った例である。

リスト 2-7-8 ソースコードのビルドコマンド

```
C:¥Users> cd C:¥vc4d-build¥vc4d                                    ソースフォルダの移動
C:¥vc4d-build¥vc4d> set JAVA_HOME=C:¥Program Files¥Java¥jdk1.7.0   環境変数を設定
C:¥vc4d-build¥vc4d> C:¥vc4d-build¥apache-maven-3.0.4¥bin¥mvn.bat package   ビルド実行
    :
    :   ビルド実行メッセージ
    :
C:¥vc4d-build¥vc4d> dir target¥vc4d.war                            vc4d.war の生成確認
```

（６）操作説明

①各種設定

1）共通設定プロパティファイル

1-1）概要

本システム全般で利用するプロパティファイル。

ファイル名は “vc4d.properties” で固定とする

本ファイルは ${TOMCAT_HOME}/webapps/vc4d/WEB-INF/classes フォルダ下に存在する。

1-2) 設定項目一覧

表 2-7-8 VC-4D システムのプロパティ設定項目一覧

項目名	内容
db.driverClassName	使用する JDBC ドライバのクラス名称。
db.url	接続先データベースの URL。
db.username	データベースのユーザ名。
db.password	データベースのパスワード。
db.platform	データベースのプラットフォーム。
db.showSql	ログに SQL を出力するかを指定。
db.generateDdl	DDL (Data Definition Language)ファイルを出力するかを指定。
file.upload.dir	アップロード受付ファイルを保存するフォルダを指定。 本フォルダ下に「年／月／日／受付 ID」サブフォルダを生成し、各ファイルを格納する。
file.download.dir	ダウンロード受付ファイルを保存するフォルダを指定。 本フォルダ下に「年／月／日／受付 ID」サブフォルダを生成し、各ファイルを格納する。
file.system.db.names	システムデータベース名称を、カンマ区切りで指定。 アップロード処理において、本項目で指定したデータベース名と重複した場合は、${dbname}_data と変更される。
job.command	仮想コンバータプログラムのパス名を指定。
job.db.overwrite	アップロード処理において、データベースを上書きするかを指定。 “false”を設定した場合、データベース名は連番を付与して作成される。
job.command.env.S	仮想コンバータ用データベースのサーバ名称。
job.command.env.U	仮想コンバータ用データベースのユーザ名。
job.command.env.P	仮想コンバータ用データベースのパスワード。
job.command.env.D	仮想コンバータ用データベースファイルの格納フォルダ。
job.cron.expression	仮想コンバータの実行時間を指定。 UNIX cron ベースの書式で、“秒 分 時 日 月 週 年”を指定する。 尚、初期設定値は一分間隔で実行されるよう設定されている。

※ 基本的に**太字**の部分が、設定変更の必要な項目

1-3) 設定例

リスト 2-14 に、共通設定プロパティファイルの設定例を記す。

形式は［項目名］＝［値］となる。

設定する値の中で、“¥”はエスケープ文字と認識されるため、“¥”を設定する場合は“¥¥”と入力する。

（Windows 環境ではファイルパス区切子として、“¥”が使用される）

また、値の中で“${項目名}”とすると変数として認識され、該当する項目の値が展開される。

行頭が“#”の行はコメントとして認識される。

リスト 2-7-9 プロパティファイル vc4d.properties の設定

```
# DBMS settings.
db.driverClassName=net.sourceforge.jtds.jdbc.Driver
db.url=jdbc:jtds:sqlserver://localhost/vc4d
db.username=sa
db.password=password
db.platform=org.hibernate.dialect.SQLServer2008Dialect
db.showSql=false
db.generateDdl=false

# File store settings.
vc4d.dir=C:¥¥vc4d
file.upload.dir=${vc4d.dir}¥¥file¥¥upload
file.download.dir=${vc4d.dir}¥¥file¥¥download
file.system.db.names=master,model,msdb,tempdb,vc4d

# File store settings.
job.command=${vc4d.dir}¥¥exe¥¥vc4d.exe
job.db.overwrite=false
job.command.env.S=(local)¥¥SQLEXPRESS
job.command.env.U=${db.username}
job.command.env.P=${db.password}
job.command.env.D=${vc4d.dir}¥¥data
job.cron.expression=0 */1 * * * ?
```

1-4) ファイルの編集／システムへの反映

テキストエディタを用いて、動作環境に沿う内容で本ファイルを編集する。

ファイル編集後に Apache Tomcat を再起動することで、システムへ反映される。

２）仮想コンバータ用メッセージファイル

2-1) 概要

仮想コンバータの終了コードより、実行結果メッセージを設定するプロパティファイル。

ファイル名は “job-messages.properties” で固定とする

本ファイルは ${TOMCAT_HOME}/webapps/vc4d/WEB-INF/classes フォルダ下に存在する。

2-2) 設定内容一覧

表 2-7-9 VC-4D システムの実行結果メッセージ

項目名	内容
code_0	仮想コンバータ 終了コード 0 に該当するメッセージ。
code_1	仮想コンバータ 終了コード 1 に該当するメッセージ。
code_2	仮想コンバータ 終了コード 2 に該当するメッセージ。
code_3	仮想コンバータ 終了コード 3 に該当するメッセージ。
code_4	仮想コンバータ 終了コード 4 に該当するメッセージ。
code_5	仮想コンバータ 終了コード 5 に該当するメッセージ。
code_6	仮想コンバータ 終了コード 6 に該当するメッセージ。
code_-1	仮想コンバータ実行前にエラーが発生した時のメッセージ。 具体的には、共通設定プロパティファイルの job.command 項目において、仮想コンバータプログラムへのパスが間違っている場合など。
code_-2	仮想コンバータ実行後、result.txt が出力されない時のメッセージ。
extCode_9	result.txt に出力される、exe_ret_code=9 に該当するメッセージ。

2-3) ファイルの編集／システムへの反映

仮想コンバータの終了コードの意味を変更した場合は、本ファイルのメッセージ内容を変更する必要がある。

また、終了コードを増やした場合は、本ファイルに対して “code_（終了コード番号）＝メッセージ内容” の形式で追加を行う。

result.txt 中の exe_ret_code の内容を追加する場合は、“extCode_（exe_ret_code の番号）＝メッセージ内容” の形式とする。

ファイル編集後に Apache Tomcat を再起動することで、システムへ反映される。

②操作画面

１）本システムの URL

任意の Web ブラウザより、以下の URL にアクセスを行う。

http://（インストールサーバのホスト名）/vc4d/

２） ３次元データアップロード画面

画面左側メニューの「アップロード受付」より、３次元データアップロード受付画面が表示される。

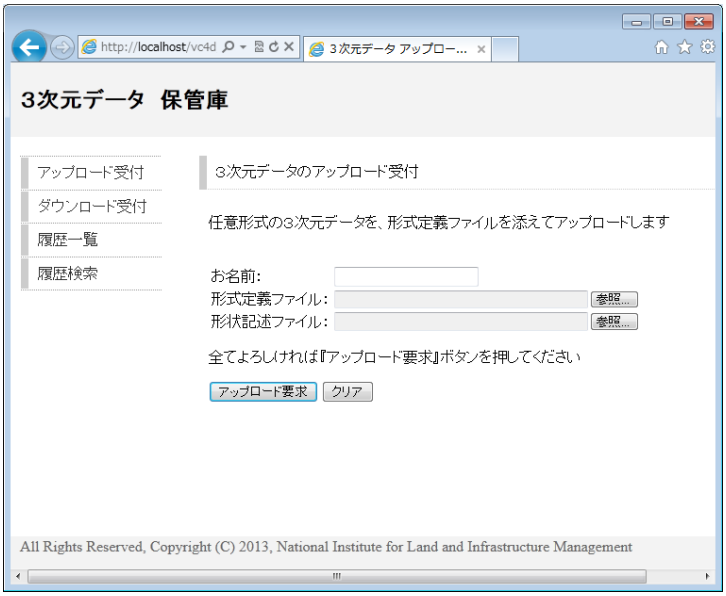


図 2-7-18 アップロード受付画面

受付用フォームに指定する内容を表 2-7-10 に示す。

表 2-7-10 VC-4D アップロード受付フォームの指定内容

項目名	内容
お名前	任意の文字列を入力する。
形式定義ファイル	3次元データの展開に利用する、メタ情報ファイルを指定する。
形状記述ファイル	3次元データの実データファイルを指定する。 仮想コンバータでは、本ファイルの拡張子を除いた名称でデータベースを作成し、3次元データを展開する。

フォームの各項目に文字列／ファイルを指定後、「アップロード要求」ボタンを押すことで、サーバ上に受け付けた内容が登録され、受付 ID が発行される。

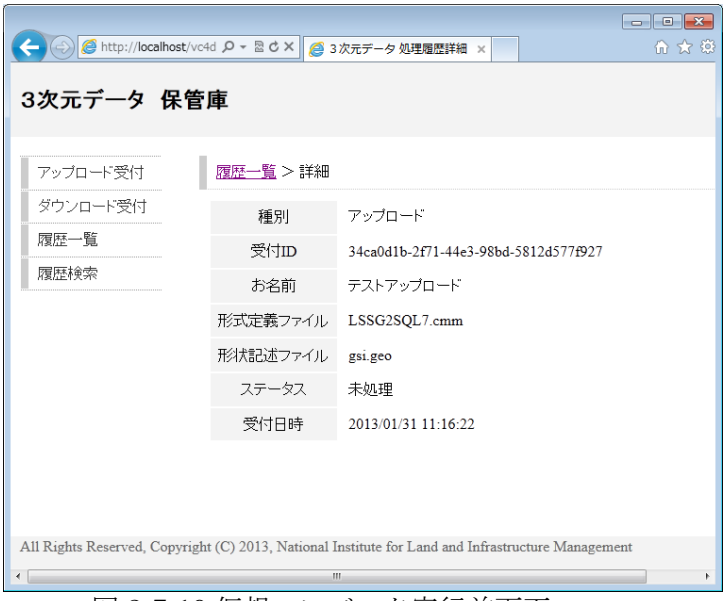


図 2-7-19 仮想コンバータ実行前画面

発行された ID は、後述する履歴検索において用いるため、ユーザは必要に応じてテキストファイル等に保存しておく必要がある。

アップロードを受け付けた後は、履歴詳細画面へ遷移する。

画面中の「ステータス」が“未処理”の場合、仮想コンバータでのデータ展開処理はまだ行われていない。「ステータス」が“未処理”または“処理中”の場合、本画面は１０秒ごとに自動リロードされる。

「ステータス」が“完了”の場合、仮想コンバータでの処理は終了しており、処理結果が表示される。

処理が成功した場合は、「登録データベース名称」に３次元データを展開したデータベース名が表示される。

図 2-7-20 仮想コンバータ実行完了画面

なお「登録データベース名称」について以下の禁則と対応を行っている。

2-1) ３次元データが展開されるデータベースの名称（登録データベース名称）には、受付画面で指定した「形状記述ファイル」のファイル名を使用し、通常は形状記述ファイルのファイル名から拡張子を除いた文字列を登録データベース名称として用いる。

2-2) 共通設定プロパティファイルにおいて、登録データベースの上書きを許可しない（job.db.overwrite=false）設定の場合は、データベース名が重複しないよう枝番号が付与される。

2-3) ファイル名に、データベース作成において使用できない記号が含まれる場合は、記号が全て‘_’に置き換えられる。

2-4) ファイル名の先頭文字が数字で始まる場合は、データベース名の先頭に‘_’が追加される。

登録データベース名称の変換例を表 2-7-11 に示す。

表 2-7-11 VC-4D 登録データベース名称の変換

形状記述ファイル名	登録データベース名称	条件	説明
gis.geo	gis	（通常）	（ファイル拡張子が除かれる）
gis.geo	gis_1	既に“gis”が登録済み	重複しないよう枝番号が付与される
gis-(new).geo	gis_new_	ファイル名に記号が含まれる	記号は‘_’に変換される
123-gis.geo	_123_gis	ファイル名の先頭文字が数字	先頭に‘_’が追加される

3） 3 次元データダウンロード画面

画面左側メニューの「ダウンロード受付」より、3 次元データダウンロード受付画面が表示される。

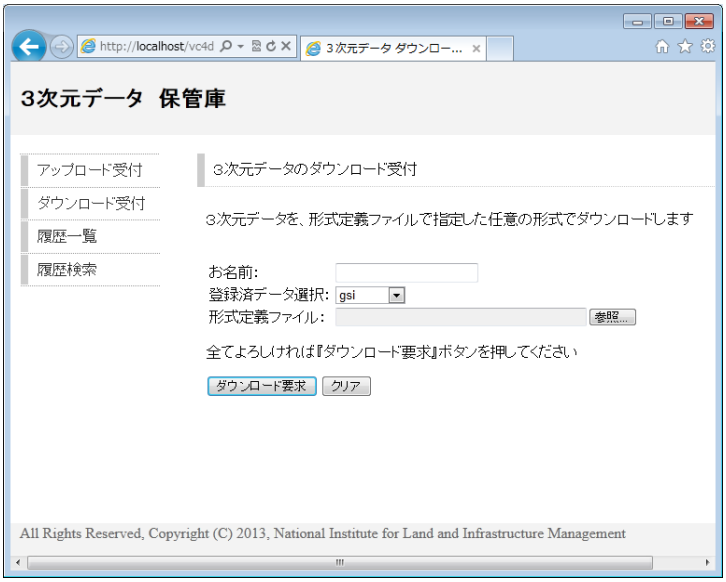


図 2-7-21 ダウンロード受付画面

受付用フォームに指定する内容を表 2-7-12 に示す。

フォームの各項目に文字列／ファイルを指定後、「ダウンロード要求」ボタンを押すことで、サーバ上に受け付けた内容が登録され、受付 ID が発行される。

発行された ID は、後述する履歴検索において用いるため、必要に応じてテキストファイル等に保存しておく。

ダウンロードを受け付けた後は、履歴詳細画面へ遷移する。

画面中の「ステータス」が“未処理”の場合、仮想コンバータでのファイル生成処理は行われていない。

表 2-7-12 VC-4D ダウンロード受付フォームの指定内容

項目名	内容
お名前	任意の文字列を入力する。
登録済データ選択	3次元データが登録されているデータベース名称を指定する。 仮想コンバータで生成する形状記述ファイル名は、ここで指定した値が用いられる。 （拡張子は形式定義ファイルで定義した文字列が付与される）
形式定義ファイル	3次元データのファイル生成に利用するメタ情報ファイルを指定する。

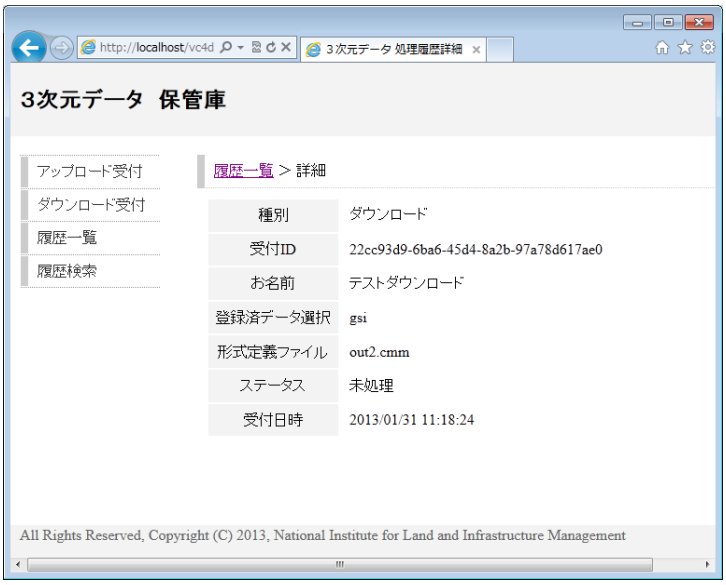


図 2-7-22 仮想コンバータ実行前画面

「ステータス」が“未処理”または“処理中”の場合、本画面は１０秒ごとに自動リロードされる。

「ステータス」が“完了”の場合、仮想コンバータでの処理は終了しており、処理結果が表示される。

処理が成功した場合は、「形状記述ファイル ダウンロード」に生成したファイル名が表示される。

ファイル名のリンクをクリックすることで、ファイルダウンロードが行える。

図 2-7-23 仮想コンバータ実行完了画面

4）履歴一覧画面

画面左側メニューの「履歴一覧」より、現在使用するブラウザよりアップロード／ダウンロード要求を行った履歴の一覧が表示される。

図 2-7-24 履歴一覧画面

履歴一覧中の「受付日時」をクリックすることで、各履歴の詳細画面へ遷移する。

図 2-7-25 履歴詳細画面

アップロード／ダウンロードの履歴は、Web ブラウザの Cookie 情報と紐付けされているため、他のブラウザまたは別環境（ＰＣ）では参照できない。

また同じブラウザであってもユーザが Cookie 情報を消去した場合は、同様に参照することはできない。（この様な場合は履歴検索より参照する必要がある）

5）履歴検索画面

画面左側メニューの「履歴検索」より、過去に登録したアップロード／ダウンロード要求を行った履歴を、受付 ID より検索する画面が表示される。

図 2-7-26 履歴検索画面

フォーム中の「受付 ID」に対して過去登録時に発行された ID 入力し、「検索」をクリックすることで該当する履歴の詳細画面へ遷移する。

図 2-7-27 履歴検索結果表示画面

（７）システムの移行

①現行環境での作業

システム移行時に、現行環境では表 2-7-13 に示した作業を行う。

表 2-7-13 VC-4D システム移行時の現行環境における作業内容

	名称	内容
1	データバックアップ	vc4d データベースおよび３次元データが展開された各データベースのエクスポート、本システムにアップロードされたファイルのバックアップを行う。
2	システムのアンインストール	本システムおよび本システムが利用する各アプリケーションのアンインストールを行う。

1）データバックアップ

1-1) データベースのバックアップ

データベースのバックアップには、SQL Server に付属する“bcp”コマンドを用いる。

“bcp”コマンドでは、各データベースのテーブル単位にファイルへエクスポートを行う。

コマンドプロンプトより、『bcp [データベース名].[スキーマ名].[テーブル名] OUT [出力ファイル名] –c –T』の形式で実行する。リスト 2-7-10 に例を示す。

リスト 2-7-10 vc4d データベースのテーブルデータをエクスポートするコマンド

```
bcp vc4d.dbo.Master OUT vc4d-Master.csv -c  -T
bcp vc4d.dbo.JobTask OUT vc4d-JobTask.csv -c  -T
```

3 次元データが展開された全てのデータベースに対して、各テーブル（COLOR, COORD, FACE, GRUPPE, LINK, NORMAL, TCOORD, VERTEX）のバックアップを実行する。

エクスポートするファイル名が重複しないよう、ファイル名は“[データベース名]-[テーブル名].csv”などとする。

リスト 2-7-11 に 3 次元データベース（=post）のテーブルデータをエクスポートする例を示す。

リスト 2-7-11 vc4d のテーブルデータをエクスポートするコマンド

```
bcp post.dbo.COLOR OUT post-COLOR.csv -c  -T
bcp post.dbo.COORD OUT post-COORD.csv -c  -T
bcp post.dbo.FACE OUT post-FACE.csv -c  -T
bcp post.dbo.GRUPPE OUT post-GRUPPE.csv -c  -T
bcp post.dbo.LINK OUT post-LINK.csv -c  -T
bcp post.dbo.NORMAL OUT post-NORMAL.csv -c  -T
bcp post.dbo.TCOORD OUT post-TCOORD.csv -c  -T
bcp post.dbo.VERTEX OUT post-VERTEX.csv -c  -T
```

1-2) アップロードファイルのバックアップ

本システムにアップロードされたファイルを圧縮してバックアップを行う。

共通設定プロパティファイルに定義した、“file.upload.dir”と“file.download.dir”以下のフォルダを、任意の圧縮／解凍ツールを用いて ZIP ファイルへ圧縮する。

2）システムのアンインストール

「(4)システムのアンインストール(2-84)」の内容に従い、現行環境より本システムおよび各アプリケーションのアンインストールを行う。

vc4d データベースおよび 3 次元データが展開された各データベースは、削除前に必ずデータエクスポートを行うこと。

②移行環境での作業

次に、移行環境で表 2-7-14 に示した作業を行う。

表 2-7-14 VC-4D システム移行時の移行環境における作業内容

	名称	内容
1	システムのインストール	本システムおよび本システムが利用する各アプリケーションのインストールを行う。
2	バックアップデータの復元	vc4d データベースおよび 3 次元データが展開された各データベースの作成／データインポート、本システムにアップロードされたファイルの展開を行う。

1）システムのインストール

「(3)システムのインストール(2-75)」の内容に従い、移行環境に本システムおよび各アプリケーションのインストールを行う。

2）バックアップデータの復元

2-1) データベースのリストア

始めに３次元データを登録するデータベースを作成する。

作成するデータベースの名称は、vc4d データベースの Master テーブルの内容より決定する（現行環境よりエクスポートしたファイルの内容を参照する)。

データベースの作成は、「0．システム用データベース／テーブルの作成」の内容を参考にして、SQL Server Manager Studio より行う。

作成後は､各データベースにテーブル（COLOR, COORD, FACE, GRUPPE, LINK, NORMAL, TCOORD, VERTEX）を作成する。リスト 2-7-12 に例を示す。

リスト 2-7-12　３次元データベース（=post）のテーブルを作成する SQL 文

```
USE [post]
GO

SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO

CREATE TABLE [dbo].[COLOR](
	[id] [int] IDENTITY(1,1) NOT NULL,
	[R] [float] NULL,
	[G] [float] NULL,
	[B] [float] NULL,
	[A] [float] NULL
) ON [PRIMARY]
GO

CREATE TABLE [dbo].[COORD](
	[id] [int] IDENTITY(1,1) NOT NULL,
	[X] [float] NULL,
	[Y] [float] NULL,
	[Z] [float] NULL
) ON [PRIMARY]
GO

CREATE TABLE [dbo].[FACE](
	[Gid] [int] NULL,
	[Fid] [int] NULL,
	[idcolor] [int] NULL,
	[idnormal] [int] NULL,
	[idmaterial] [int] NULL,
	[idtexture] [int] NULL,
	[SHP] [int] NULL
) ON [PRIMARY]
GO

CREATE TABLE [dbo].[GRUPPE](
	[Gid] [int] NULL,
	[idmaterial] [int] NULL,
	[idtexture] [int] NULL,
	[Attribute] [varchar](max) NULL
) ON [PRIMARY]
GO

CREATE TABLE [dbo].[LINK](
	[id] [int] IDENTITY(1,1) NOT NULL,
	[GPid] [int] NULL,
	[GCid] [int] NULL,
	[mat0] [float] NULL,
	[mat1] [float] NULL,
	[mat2] [float] NULL,
```

```
	[mat3] [float] NULL,
	[mat4] [float] NULL,
	[mat5] [float] NULL,
	[mat6] [float] NULL,
	[mat7] [float] NULL,
	[mat8] [float] NULL,
	[mat9] [float] NULL,
	[mat10] [float] NULL,
	[mat11] [float] NULL,
	[mat12] [float] NULL,
	[mat13] [float] NULL,
	[mat14] [float] NULL,
	[mat15] [float] NULL
) ON [PRIMARY]
GO

CREATE TABLE [dbo].[NORMAL](
	[id] [int] IDENTITY(1,1) NOT NULL,
	[X] [float] NULL,
	[Y] [float] NULL,
	[Z] [float] NULL
) ON [PRIMARY]
GO

CREATE TABLE [dbo].[TCOORD](
	[id] [int] IDENTITY(1,1) NOT NULL,
	[X] [float] NULL,
	[Y] [float] NULL
) ON [PRIMARY]
GO

CREATE TABLE [dbo].[VERTEX](
	[Gid] [int] NULL,
	[Fid] [int] NULL,
	[Ix] [int] NULL,
	[Icoord] [int] NULL,
	[Inormal] [int] NULL,
	[Icolor] [int] NULL,
	[Itcoord] [int] NULL
) ON [PRIMARY]
GO
```

各テーブルのデータインポートには、SQL Server に付属する“bcp”コマンドを用いる。

コマンドプロンプトより、『bcp [データベース名].[スキーマ名].[テーブル名] IN [入力ファイル名] –c –T』の形式で実行する。

入力ファイルは、現行環境よりエクスポートした各データベースとテーブルに対応するファイルを指定する。リスト 2-7-13、14 にコマンドの例を示す。

リスト 2-7-13　3 次元データベースのテーブルデータをインポートするコマンド

（データの名称：post）

```
bcp post.dbo.COLOR IN post-COLOR.csv -c   -T
bcp post.dbo.COORD IN post-COORD.csv -c   -T
bcp post.dbo.FACE IN post-FACE.csv -c   -T
bcp post.dbo.GRUPPE IN post-GRUPPE.csv -c   -T
bcp post.dbo.LINK IN post-LINK.csv -c   -T
bcp post.dbo.NORMAL IN post-NORMAL.csv -c   -T
bcp post.dbo.TCOORD IN post-TCOORD.csv -c   -T
bcp post.dbo.VERTEX IN post-VERTEX.csv -c   -T
```

リスト 2-7-14 vc4d データベースのテーブルデータをインポートするコマンド

```
bcp vc4d.dbo.Master IN vc4d-Master.csv -c   -T
bcp vc4d.dbo.JobTask IN vc4d-JobTask.csv -c   -T
```

2-2) アップロードファイルの展開

現行環境で作成した ZIP ファイル（アップロードファイルのバックアップ）を、所定のフォルダ下で展開する。

尚、展開先は現行環境と同じフォルダとする。