IIS と ISAPI REWRITE

By Timothy Penner, Technical Services Team Member, 4D Inc.

Technical Note 09-27

概要

このテクニカルノートでは、4D Web サーバのフロントエンドとして Microsoft IIS を使用する方法について議論します。これはサードパーティーの ISAPI フィルタである Helicon Tech 社の ISAPI_Rewrite を使用します。いくつかのサンプル設定を紹介し、それぞれのオプションについて議論します。この情報は、顧客の求めに応じて 4D のデベロッパが 4D Web サーバのフロントエンドに IIS を使用するための必要な知識を提供します。

はじめに

サードパーティーの ISAPI フィルタである Helicon Tech 社の ISAPI_Rewrite を使用することで、4D Web サーバのフロントエンドとして Microsoft ISS を使用することがとても簡単になります。このテクニカルノートでは様々な ISAPI_Rewrite の設定を試し、利用可能なオプションについて見ていくことにします。この情報は、顧客の求めに応じて 4D のデベロッパが 4D Web サーバのフロントエンドに IIS を使用するための必要な知識を提供します。

語句について

IIS は “Internet Information Services” の略であり、最近の Microsoft OS に含まれる Microsoft 社の Web サーバです。IIS は ISAPI として知られる API を使用します。ISAPI は “Internet Server Application Programming Interface”の略で、Microsoft IIS の API です。

ISAPI_Rewrite は Helicon Tech 社が作成した ISAPI フィルタであり、IIS サーバ上で特定のリクエストをキャッチし、それらのリクエストを (4D Web サーバなど) 他のサーバに送信して、レスポンスをリクエスト者にプロクシするために使用します。
Helicon Tech 社は ISAPI_Rewrite フィルタについて以下のように述べています:

ISAPI_Rewrite は正規表現に基づき URL を処理するパワフルなエンジンです。これは Microsoft 社の Internet Information Server (IIS) 用ですが、ほとんど Apache の mod_Rewrite のように動作します。ISAPI_Rewrite は C/C++で書かれた ISAPI フィルタであり、とても速く動作します。ISAPI_Rewrite を使用して、標準の URL スキームを飛び出し、あなた独自のスキームを開発できます。

ソース: http://www.isapirewrite.com/ (製品概要)

ISAPI_Rewrite フィルタは Helicon Tech 社から入手できます
アドレス: http://www.helicontech.com/download-ISAPI_Rewrite3.htm

使用する理由

Microsoft IIS は業界標準の Web サーバであり、多くの組織が利用しています (主な理由は IIS が Microsoft 社のサーバ OS に含まれているためです)。多くの企業の IT 部署では IIS の利用に満足していて、そして彼ら自身が満足するものを採用したいと考えています。このことは時に、大企業のプロジェクトに対する 4D 組み込みの Web サーバの提案を難しくします。IT の人は彼が知っているものを採用したがるのです。

幸運なことにサードパーティーの ISAPI フィルタの助けを借りることで、IIS を 4D Web サーバのフロントエンドとして利用し、IT の人が彼らが知る製品をそのまま使用しつつ、動的 (あるいは静的な) コンテンツを 4D Web サーバから配信することが可能です。

ISAPI_Rewrite フィルタを使用して、以下のことが可能です:

- 動的なコンテンツに対し検索エンジン最適化を行う
- 他サイトからあなたのサイトのファイルへのホットリンクをブロックする
- 独自の認証スキームを構築し、独自のスクリプトやデータベースを使用してスタティックファイルへのアクセスを管理する
- あるサイトのコンテンツを他のサイトのディレクトリにプロクシする
- イントラネットサーバを構築する
- 物理的な一つのサイトで、HOST ヘッダベースのダイナミックホストを作成する
- スタティックページに対してもブラウザごとに異なるページを配信する
- その他多くの課題を、ISAPI_Rewrite フィルタに内蔵された正規表現エンジンを使用して解決できます。

このテクニカルノートは主に 4D Web サーバのフロントエンドとして IIS を使用する (リバースプロクシ) ことにフォーカスします。

どのように動作するか

ISAPI_Rewrite フィルタは IIS のエクステンションであり、4D の拡張ではありません。つまりこの拡張は IIS Web サーバと同じマシンにインストールされます (ダウンロードリンク: http://www.helicontech.com/download-ISAPI_Rewrite3.htm)。IIS と 4D Web サーバが同じマシンにインストールされているのでない限り、フィルタを 4D Web サーバマシンにインストールする必要はありません。もちろん 4D Web サーバと IIS が同じマシン上にインストールされていない場合、設定で指定されたポートを通じてお互いに正しく通信できなければなりません。

Note: 4D 側に追加の設定は必要ありません。IIS マシンに ISAPI フィルタをインストールし、4D Web サーバにリクエストが転送されるよう設定を行うだけです。

典型的な例では IIS Web サーバが標準ポート 80 で動作し、4D Web サーバは非標準ポート(例えば 8080) で動作します。そして ISAPI_Rewrite は特定条件のリクエストをキャッチし、4D Web サーバにそのリクエストをプロクシするよう設定されます。

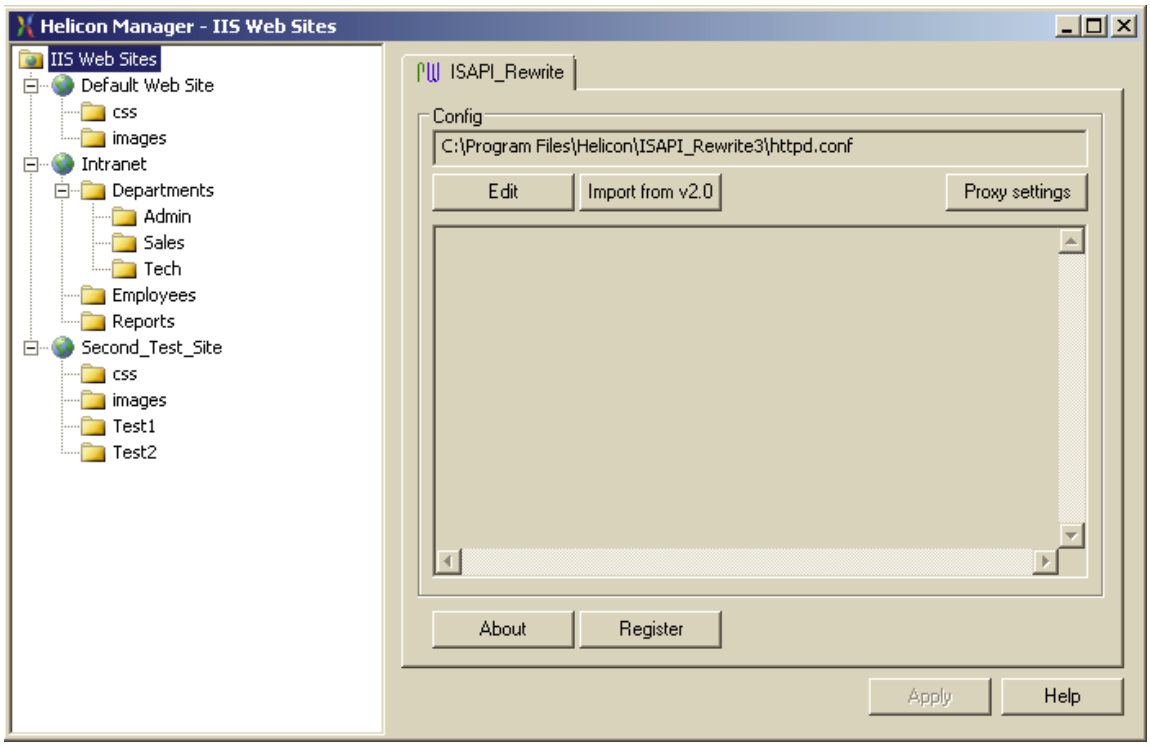
ISAPI_Rewrite フィルタは正規表現エンジンを使用して、Apache の mod_proxy と同様の能力を IIS の世界にもたらします。設定は 2 つの異なるカテゴリに分けられます:

- すべてのサイトに適用するグローバル設定。これはアプリケーションのインストール先に格納され、デフォルトは C:\Program Files\Helicon\ISAPI_Rewrite3\httpd.conf です。
- 個々のサイトおよび内部のディレクトリ固有の設定。これらの設定ファイルは.htaccess として格納され、設定が適用されるディレクトリに置かれます。

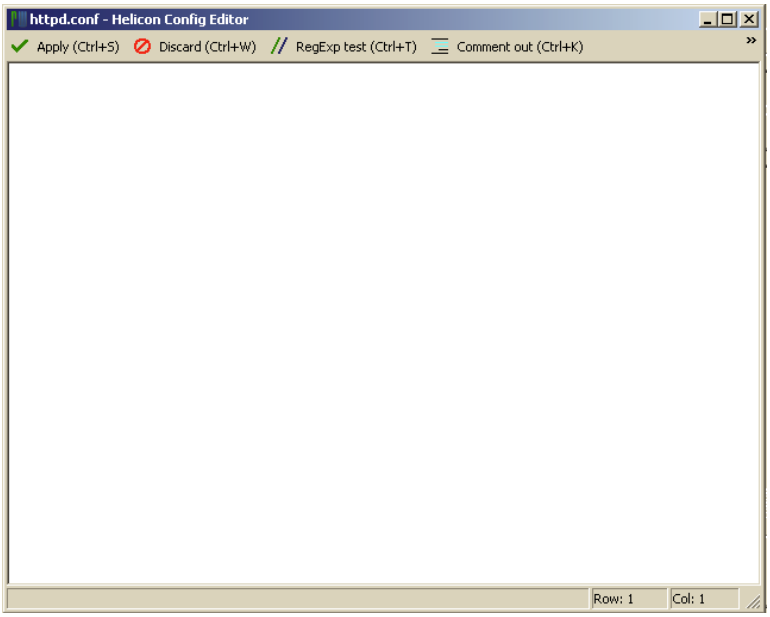
設定内容は Helicon Manager (または ISAPI_Rewrite manager とも呼ばれる) を通して見ることができます。

Helicon ISAPI REWRITE Manager アプリケーションを使用する

ISAPI_REWRITE フィルタには管理アプリケーションが含まれていて、設定を行うために使用できます。以下の図は最初に Helicon Manager を起動したときの様子です:



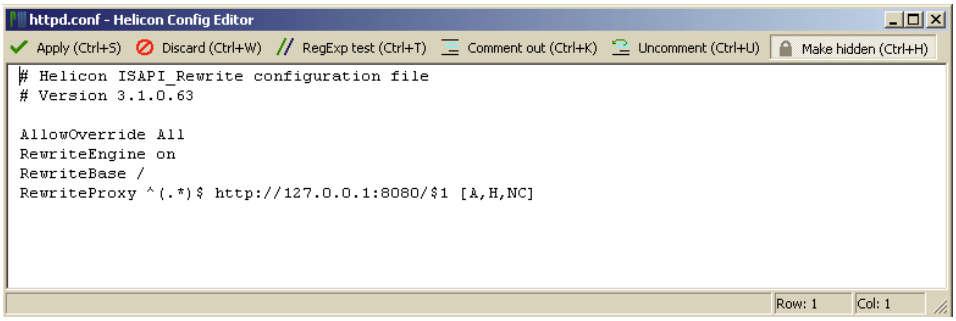
このテクニカルノートでは全体の設定ファイルに対してすべての設定を行います。設定ファイルを編集するには、まず編集を行うレベルを選択し (例: 全体の設定を行うには“IIS Web Sites”アイコンを選択します)、そして”Edit”ボタンをクリックします。すると編集ウィンドウが表示されます:



Helicon Config Editor を使用して、特定の設定を入力また更新できます。以下の設定例を入力してみましょう:

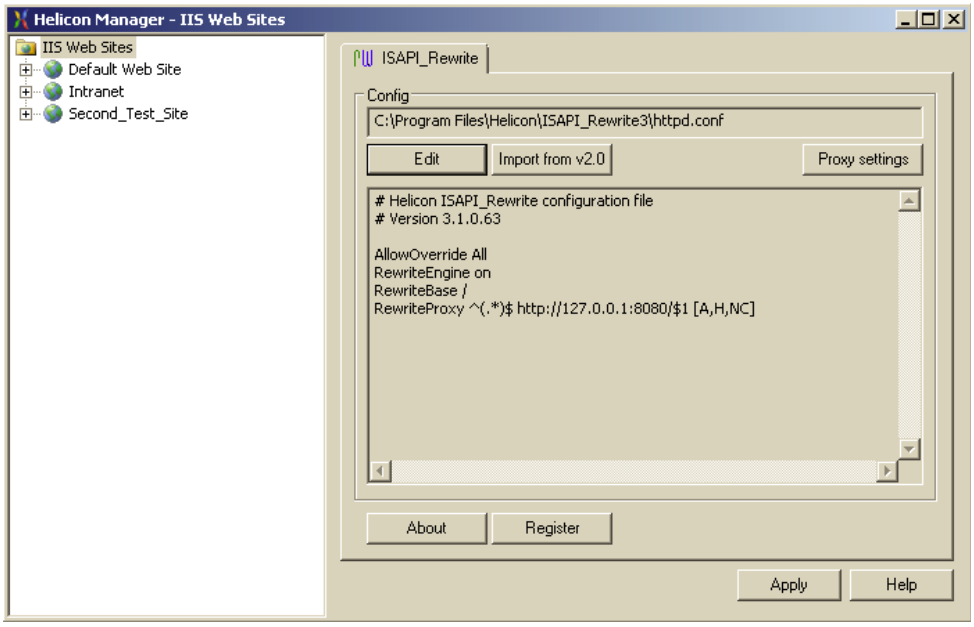
```
# Helicon ISAPI_Rewrite configuration file
# Version 3.1.0.63

AllowOverride All
RewriteEngine on
RewriteBase /
RewriteProxy ^(.*)$ http://127.0.0.1:8080/$1 [A,H,NC]
```

タイプした結果は以下のようになるはずです:



このウィンドウの編集が終わったら、ツールバー上の  ボタンをクリックします。

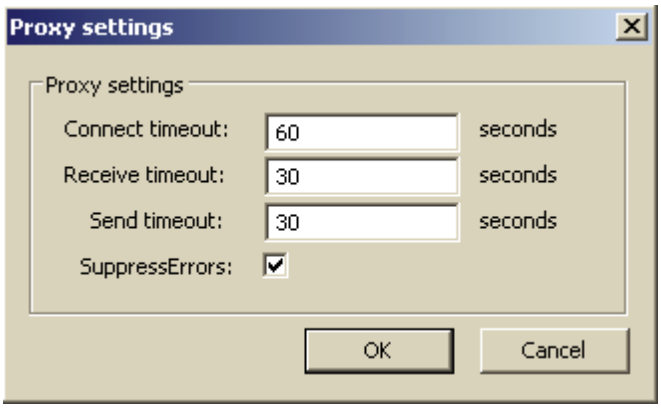
Apply をクリックすると、設定ファイルに変更内容が保存され、Helicon Manager に戻ります;



しかし行った更新は Helicon Manager の Apply ボタンをクリックするまで有効にはなりません。

プロクシ設定の変更

さらに Helicon Manager は IIS administrator を使用して“Proxy setting” ボタンで利用できるプロクシタイムアウト設定 の変更を可能にします。設定オプションには接続タイムアウト (デフォルト 60 秒)、受信タイムアウト (デフォルト 30 秒)、送信タイムアウト (デフォルト 30 秒)、そしてエラー削除オプション (デフォルトで有効) が含まれます。



Note: *デバッガを使用したコードの検証時には通常これらのタイムアウトより多くの時間が必要です。アプリケーションの開発中にはこれらのタイムアウト値を増やして、タイムアウトが原因の問題を発生させないようにすることをお勧めします。*

このテクニカルノートで使用されるディレクティブ

この節では、このテクニカルノートで使用されるディレクティブについて説明します。ディレクティブの完全なリストおよびその利用方法は HeliconTech 社の ISAPI_Rewrite ドキュメントページを参照してください:
http://www.helicontech.com/isapi_rewrite/doc/

LogLevel

```
ディレクティブ: LogLevel
説明: 一般的なエラーのログレベルを設定
シンタックス: LogLevel Level
デフォルト: LogLevel warn
コンテキスト: server config
```

LogLevel ディレクティブはリライト処理に関係なく、一般的なログ出力の冗長性を設定します。

以下は LogLevel ディレクティブの Level パラメタに指定可能な値のリストです:

- Emerg
- Alert
- Crit
- Error
- Warn
- Notice
- Info
- debug

現在 ISAPI_Rewrite ログは debug と crit エラーメッセージのみを記録します。

LogLevel を debug に設定すると、設定ファイルのロードに関するトラブルシュートができます。

LogLevel ディレクティブの完全な説明は以下のサイトを参照してください:
http://www.helicontech.com/isapi_rewrite/doc/LogLevel.htm

RewriteLogLevel

```
ディレクティブ: RewriteLogLevel
説明: ログのレベルを設定
シンタックス: RewriteLogLevel Level
デフォルト: RewriteLogLevel 0
コンテキスト: server config
```

RewriteLogLevel ディレクティブはログ出力の冗長性を設定します。デフォルト値は 0 でログは記録されません。最大値は 9 ですべてのアクションがログに記録されます。

高い値を設定すると ISAPI_Rewrite の動作が遅くなります。ルールセットのデバッグが完了したら、値を 0 に設定してログを無効にすることをお勧めします。

RewriteLogLevel ディレクティブの完全な説明は以下のサイトを参照してください:
http://www.helicontech.com/isapi_rewrite/doc/RewriteLogLevel.htm

RewriteLog

```
ディレクティブ: RewriteLog
説明: ISAPI_Rewrite ログファイルの名称を設定する
シンタックス: RewriteLog file-path
デフォルト: RewriteLog installdir\rewrite.log
コンテキスト: server config
```

RewriteLog ディレクティブは、ISAPI_Rewrite がアクションを記録するログファイルの場所と名称を設定します。例:

RewriteLog "C:\local\path\rewrite.log"

RewriteLog ディレクティブの完全な説明は以下のサイトを参照してください:
http://www.helicontech.com/isapi_rewrite/doc/RewriteLog.htm

ErrorLog

```
ディレクティブ: ErrorLog
説明: 一般的エラーのファイルの場所
シンタックス: ErrorLog file-path
デフォルト: ErrorLog installdir\rewrite.log
コンテキスト: server config
```

ErrorLog ディレクティブは、ISAPI_Rewrite が一般的なエラーエラーおよび httpd.conf ファイルや.htaccess ファイルのロードのメッセージを記録するログファイルの場所と名称を設定します。例:

ErrorLog "C:\local\path\error.log"

ErrorLog ディレクティブの完全な説明は以下のサイトを参照してください:
http://www.helicontech.com/isapi_rewrite/doc/ErrorLog.htm

AllowOverride

```
ディレクティブ: AllowOverride
説明: 各バーチャルホストやディレクトリ毎の設定を有効にするか無効にするか指定する
シンタックス: AllowOverride All|None|directive-type [directive-type] ...
デフォルト: AllowOverride All
コンテキスト: server config, virtual host, directory
```

AllowOverride ディレクティブは、配置された **.htaccess** ファイル中のディレクティブが親レベル (**httpd.conf**) のディレクティブを上書きできるかどうかを設定します。ISAPI_Rewrite においてこのディレクティブは実際には、特定のバーチャルホスト (Web サイト) やディレクトリ用の.htaccess ファイルを有効または無効にします。現在以下の 3 つの値のみがサポートされています: **All**, **None** そして **FileInfo**.

All と **FileInfo** は.htaccess ファイルとその中のすべての ISAPI_Rewrite ディレクティブを有効にします。**None** はすべての.htaccess ファイルとその中のディレクティブを無効にします。このディレクティブは継承されます。つまりあるディレクトリやバーチャルホストの.htaccess に

AllowOverride none を設定するとすべてのサブディレクトリで.htaccess ファイルが無効になります。

AllowOverride ディレクティブの完全な説明は以下のサイトを参照してください:
http://www.helicontech.com/isapi_rewrite/doc/AllowOverride.htm

RewriteEngine

```
ディレクティブ: RewriteEngine
説明: ランタイムリライトエンジンを有効または無効にする
シンタックス: RewriteEngine on|off
デフォルト: RewriteEngine off
コンテキスト: server config, virtual host, directory, .htaccess
```

RewriteEngine ディレクティブはリライトランタイムを有効または無効にします。
ISAPI_Rewrite モジュールまたは特定の.htaccess ファイルを無効にする必要がある場合、リライトルールのコメントアウトではなく、RewriteEngine off を使用してください。

警告! *クエリのサポートのため、HeliconTech 社はデフォルトでリライトエンジンの有効にする必要がありました。対して Apache のデフォルトは無効です。この点を考慮に入れ、有効にするにせよ無効にするにせよ、各設定ファイル内で明示的に設定を行ってください。*

RewriteEngine ディレクティブの完全な説明は以下のサイトを参照してください:
http://www.helicontech.com/isapi_rewrite/doc/RewriteEngine.htm

RewriteBase

```
ディレクティブ: RewriteBase
説明: ディレクトリ毎のリライトのベース URL を明示的に設定する
シンタックス: RewriteBase URL-path
デフォルト: RewriteBase requested-directory-path
コンテキスト: directory, .htaccess
```

RewriteRule ディレクティブがディレクトリ毎の設定ファイル(.htaccess) で使用されるとき、パスから自動でローカルディレクトリ接頭辞が取り除かれ、そして残りの部分にルールが適用

されます。**RewriteBase** ディレクティブを使用すると、明示的にルールで使用するベースを指定できます。例: 取り除く部分の指定など

Apache の mod_rewrite と異なり、ISAPI_Rewrite は物理的なパスだけでなくバーチャルパスへのアクセスも持っていて、自動で正しいベースを選択できます。それでこのディレクティブは互換性の理由で保持されています。

URL-path はルートの相対パスか空です。空の URL-path 値はベースが Web サイトのルートであることを意味します。

RewriteBase ディレクティブの完全な説明は以下のサイトを参照してください:
http://www.helicontech.com/isapi_rewrite/doc/RewriteBase.htm

RewriteProxy

```
ディレクティブ: RewriteProxy
説明: 外部サーバへのプロクシリクエスト
シンタックス: RewriteProxy Pattern Substitution [flags]
コンテキスト: server config, virtual host, directory, .htaccess
```

RewriteProxy ディレクティブにより、結果の URL が内部的に他のサーバのターゲットとして扱われ、即座に (例えばルールの処理はここで停止されます) リモートサーバに渡されるようになります。そしてリモートサーバのレスポンスがクライアントに返されます。Proxy は完全な URL を要求します。プロトコルから開始され、ホスト名、と続きます。ISAPI_Rewrite は ISAPI エクステンションを使用してプロクシリクエストを処理します。

シンタックスと動作は RewriteRule ディレクティブと同じですが、RewriteProxy ディレクティブは追加のフラグをサポートします:

H (preserve **H**ost)

H フラグは、リモートサーバに接続する際、プロクシモジュールがオリジナルのリクエストに含まれる Host ヘッダを使用することを意味します。このフラグがない場合、プロクシモジュールはリモートサーバのホスト名とポートから Host ヘッダを構築します。

A (**A**dd authentication headers)

A フラグを指定すると、プロクシサーバに対するクライアント認証が使用されているとき、プロクシから内部サーバに認証情報を渡すことを許可します。

プロクシされたサーバにリクエストを送信するために、プロクシモジュールはヘッダを追加します

```
X-ISRW-Proxy-AUTH-TYPE,
X-ISRW-Proxy-AUTH-USER,
X-ISRW-Proxy-LOGON-USER,
X-ISRW-Proxy-REMOTE-USER
```

対応するサーバ変数

```
AUTH_TYPE,
AUTH_USER,
LOGON_USER,
REMOTE_USER
```

CR (use **Cr**edentials)

CR フラグは URL またはベーシンク認証ヘッダで指定された認証情報を使用して、プロクシモジュールがリモートサーバへのログインを試行することを意味します。このフラグで URL に http://user:password@host.com/path/ のシンタックスが使用できます。

RewriteProxy ディレクティブの完全な説明は以下のサイトを参照してください:
http://www.helicontech.com/isapi_rewrite/doc/RewriteProxy.htm

RewriteRule ディレクティブの完全な説明は以下のサイトを参照してください:
http://www.helicontech.com/isapi_rewrite/doc/RewriteRule.htm

例題

Helicon ISAPI_REWRITE フィルタを使用して、開発者は独自の設定ファイルを記述することができ、完全にカスタマイズが可能です。この節では 4D デベロッパが興味ありまた最も使われるであろう設定について見ていくことにします。

あなたの環境においてこのアプローチのテストを行う間のデバッグの間だけでも、Helicon ISAPI_Rewrite フィルタに組み込みのログ機能を使用することをお勧めします。以下はログを有効にするための設定です:

```
# Helicon ISAPI_Rewrite configuration file
# Version 3.1.0.63

LogLevel debug
RewriteLogLevel 9

RewriteLog C:\Program Files\Helicon\ISAPI_Rewrite3\rewrite.log
ErrorLog C:\Program Files\Helicon\ISAPI_Rewrite3\error.log
```

上記の設定はデバッグログとリライトログを有効にして、両ファイルの保存先を指定しています。簡潔に見せるためにこの情報は最初の例題のみに含まれていますが、すべての例題で使用できます。

特定のディレクトリのみを 4D にプロクシする

この節では Helicon ISAPI_REWRITE フィルタを使用して IIS への特定のディレクトリを 4D からのコンテンツでプロクシする方法を示します。

```
# Helicon ISAPI_Rewrite configuration file
# Version 3.1.0.63

LogLevel debug
RewriteLogLevel 9

RewriteLog C:\Program Files\Helicon\ISAPI_Rewrite3\rewrite.log
ErrorLog C:\Program Files\Helicon\ISAPI_Rewrite3\error.log

AllowOverride All
RewriteEngine on
RewriteBase /
RewriteProxy ^4D/(.*)$ http://127.0.0.1:8080/$1 [A,H,NC]
```

この設定ファイルでは、**4D/** ではじまるすべての IIS へのリクエストをキャッチして、リクエストを 4D Web サーバが実行されているアドレスとポートである 127.0.0.1:8080 にプロクシします。このようにすると、4D はリクエストが 4D に直接行われたかのように、Web リクエストを受け取ります。この設定では、http://mysite.com/4D/ が 4D Web サーバのルートとして考えることができます。例えば:
http://mysite.com/4D/reports/sales.shtml のリクエストは 4D に
http://127.0.0.1:8080/reports/sales.shtml として送られます。

Note: この例題では URL 中の/4D/ が IIS により使用されるエントリポイントとなっていて、4D に送信されるリクエストからは取り除かれています。

SOAP リクエストのみを 4D にプロクシする

この節では Helicon ISAPI_REWRITE フィルタを使用して、SOAP リクエストを 4D Web サーバにプロクシする方法を示します。

```
# Helicon ISAPI_Rewrite configuration file
# Version 3.1.0.63

AllowOverride All
RewriteEngine on
RewriteBase /
RewriteProxy ^4DSOAP(.*)$ http://127.0.0.1:8080/4DSOAP/$1 [A,H,NC]
```

この設定ファイルでは、**4DSOAP** ではじまるすべての IIS へのリクエストをキャッチして、リクエストを 4D Web サーバが実行されているアドレスとポートである
http://127.0.0.1:8080/4DSOAP/にプロクシします。このようにすると、4D は SOAP リクエストが 4D に直接行われたかのように、リクエストを受け取ります。この設定では 4DSOAP から始まる SOAP リクエストのみが 4D Web サーバにリダイレクトされます。

SOAP リクエストおよび特定のディレクトリを 4D にプロクシする

ここでは Helicon ISAPI_REWRITE フィルタを使用して SOAP リクエストを 4D にプロクシし、IIS 上の特定のディレクトリを 4D からのコンテンツでプロクシする方法を示します。

```
# Helicon ISAPI_Rewrite configuration file
# Version 3.1.0.63

AllowOverride All
RewriteEngine on
RewriteBase /
RewriteProxy ^4D/(.*)$ http://127.0.0.1:8080/$1 [A,H,NC]
RewriteProxy ^4DSOAP(.*)$ http://127.0.0.1:8080/4DSOAP/$1 [A,H,NC]
```

この設定ファイルでは、4D/または 4DSOAP から始まるすべての IIS へのリクエストがキャッチされ、127.0.0.1:8080 で実行されている 4D Web サーバにプロクシされます。

- リクエストが 4D/で始まっている場合、リクエストは 4D Web サーバのルートにプロクシされ、 4D が受信するリクエストから “4D/” は取り除かれます。
- リクエストが 4DSOAP から始まる場合、リクエストが直接 4D Web サーバに対して行われたかのように 4D Web サーバにプロクシされます。

この方法で、4DSOAP あるいは 4D から始まるリクエストが 4D Web サーバにプロクシされます。

すべてのリクエストを 4D にプロクシする

ここでは Helicon ISAPI_REWRITE フィルタを使用して 4D からのコンテンツをすべてプロクシする方法を示します。

```
# Helicon ISAPI_Rewrite configuration file
# Version 3.1.0.63

AllowOverride All
RewriteEngine on
RewriteBase /
RewriteProxy ^(.*)$ http://127.0.0.1:8080/$1 [A,H,NC]
```

この設定ファイルでは、IIS へのリクエストがキャッチされ、127.0.0.1:8080 で実行されている 4D Web サーバにプロクシされます。この方法で 4D は IIS に送信されたすべてのリクエストを取得します。IIS Web サーバのルートを 4D Web サーバのルートと考えることができます。例えば http://mysite.com/reports/sales.shtml は http://127.0.0.1:8080/reports/sales.shtml となります。

WSDL ファイルに関する考察

4DWSDL ファイルの内容をプロクシすることは可能ですが、リライトエンジンは正しくポート番号を削除しないようです。これは WSDL ファイルを元にした Web サービスプロクシメソッドを作成する際に問題となります。このため、4DWSDL ファイルを保存し、あなたのデータベースが提供する Web サービスごとに“soap:address”要素の location 属性を修正することをお勧めします。これを行うには Web ブラウザで 4DWSDL ファイルをロードし、ファイル -> 別名で保存を選択します。これにより XML フォーマットでファイルが保存されるので、このファイルを Web サイトに配置し、Web サービスに接続する人に新しいファイルのアドレスを通知します。または修正したファイルを送信することもできます。

まとめ

このテクニカルノートでは Helicon ISAPI_Rewrite フィルタを使用して、4D からのコンテンツを IIS にプロクシする際の一般的なコンセプトについて説明しました。このテクニカルノートではフルバージョンのフィルタを使用し、様々な設定をお見せしました。この情報を使用して、4D デベロッパは顧客の求めに応じ、4D Web サーバのフロントエンドとして IIS を使用できます。

Helicon 社の製品に関する詳細情報、および ISAPI_Rewrite 3.x フィルタのライセンス情報については以下のサイトを参照してください:
http://www.helicontech.com/ISAPI_Rewrite/

ISAPI_Rewrite 3.x ダウンロード:
http://www.helicontech.com/download-ISAPI_Rewrite3.htm

ISAPI_Rewrite 3.x のサポートフォーラム:
http://www.helicontech.com/forum/forum_topics-FID-10.htm

ISAPI_Rewrite 3.x のドキュメント:
http://www.helicontech.com/isapi_rewrite/doc/