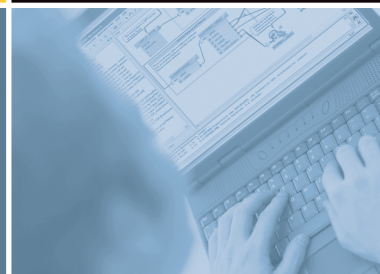


2009 FME International User Conference: Training Module
Introduction to FME Server



01. Introduction	2
Workshop Prerequisites	2
02. Overview of FME Server	3
Example – Run FME Server.....	4
03. FME Server Roles	6
User.....	6
Author.....	6
Administrator	6
FME Server & FME Workbench – A Server Client Pair	7
04. Downloading a Workspace from FME Server	8
Example - Downloading a Workspace	8
05. Published Parameters and FME Server.....	10
Published Parameters and FME Server.....	10
06. Publishing a Workspace to FME Server	11
07. What is a Service?	13
What is a Service?	13
Why use a Service?	13
FME Server Services	13
How to Use a Service.....	13
08. Data Download Service.....	15
What is a Data Download Service?.....	15
Example – Data Download.....	16
09. Administration Control Panel.....	22
Example – Managing Repositories and Workspaces.....	22
10. Architectural Considerations	23
Components FME Server.....	23
Individual Components in FME Server Installer	24
11. Data Streaming Example	25
What Formats Can be Streamed?.....	25
Example – Data Streaming	25

01. Introduction

This hands-on workshop will illustrate the distribution of data using the FME Spatial ETL (Extract, Transform, and Load) tools run through FME Server.

You will see the full life-cycle of working with FME Server: downloading a workspace, modifying it and the re-publishing it back to FME Server. You will then go on to look at the services shipped with FME Server, with a specific focus on the data download service where you will create your own data download Google Maps application. You will then look at the administration tools shipped with FME Server and the system architecture.

The workshop configuration consists of:

- FME Desktop
- FME Server
- Tomcat acting as both a web server and servlet container running on port 80

Workshop Prerequisites

Workshop attendees are assumed to have a basic knowledge of the FME Desktop, especially FME Workbench.

02. Overview of FME Server



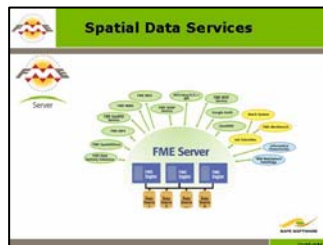
Where does it fit in?

FME Server brings the power of Safe Software's proven spatial data translation, transformation and integration technology from FME Desktop to enterprise server environments, enabling organizations to take advantage of:

- Flexible spatial data distribution
- Scalable data loading and conversion

With FME Server, your organization can address diverse spatial data requirements using a single enterprise solution. FME Server is also available in a SpatialDirect Edition that includes all of the functionality provided in SpatialDirect 2007.

Spatial Data Services



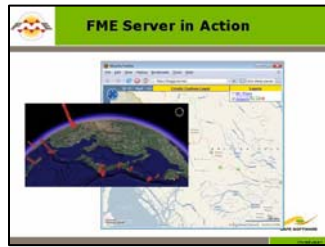
FME Server can provide many spatial data services

FME Server also offers a flexible set of spatial data services for:

- Web-based spatial data access: downloading and streaming
- Scalable data consolidation: loading and migration
- Online quality assurance: spatial data uploading and validation
- Server-based spatial data conversion: translation and transformation

Its scalable, services-oriented architecture (SOA) and ever-growing support for the latest data formats help you easily expand the system as your requirements grow and evolve.

FME Server in Action



Demonstration of FME Server

Example – Run FME Server

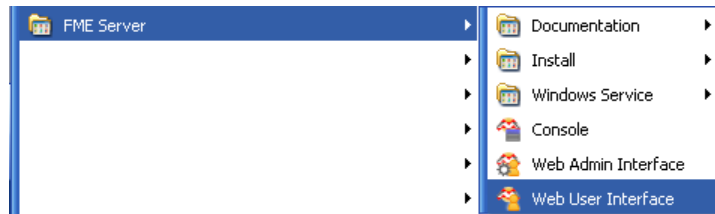
1) Open Tomcat

Open Firefox and go to <http://fmeuc2009/>. Apache Tomcat’s default webpage should open which means it is running. Apache Tomcat is used as a web server to serve the FME Server web pages and as a servlet engine to run the FME Server services.

2) Open FME Server Web User Interface

A web user interface has been built which allows users to instantly harness the power of FME Server and begin downloading and streaming data over the web. The interface also provides a starting point for developers who want to integrate the power of FME Server into their own web applications.

Navigate to *Start Menu* → *FME Server* → *Web User Interface*



Right: Accessing the Web User Interface from the Start menu

This is FME Server’s interface for submitting a request against a service.

Services

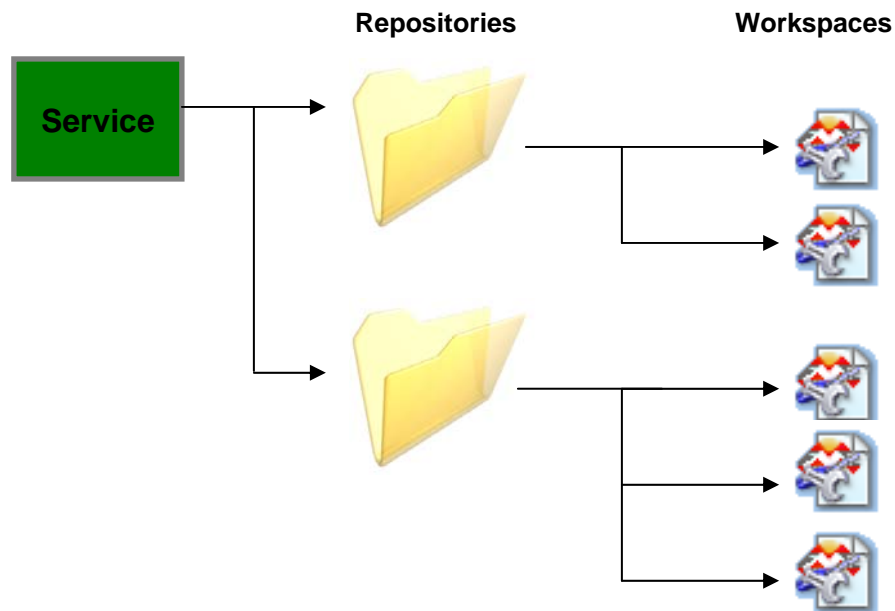
	Service / Repository / Workspace	Title	Actions
Service →	Data Download		Browse
Repository →	Samples		Browse
Workspace →	airports.fmw	BC and Alberta Airports: FFS to KML	Configure Run
	austinDownload.fmw	DataDownloadService	Configure Run
	earthquakesextrusion.fmw	Earthquakes: GeoRSS to KML Diagrams	Configure Run
	textextruder.fmw	3D Text Extruder	Configure Run
	traffic.fmw	Refractor Traffic Data: GeoRSS to KML	Configure Run
	1 repository in Data Download		

Above: Structure of FME Server Web UI

3) Services Available

Notice the six different services which FME Server can potentially provide to the user. Each one serves the data back to the user slightly differently. We will explore this later.

Each service can contain many repositories and each repository can contain many workspaces..



Above: Structure of FME Server

03. FME Server Roles

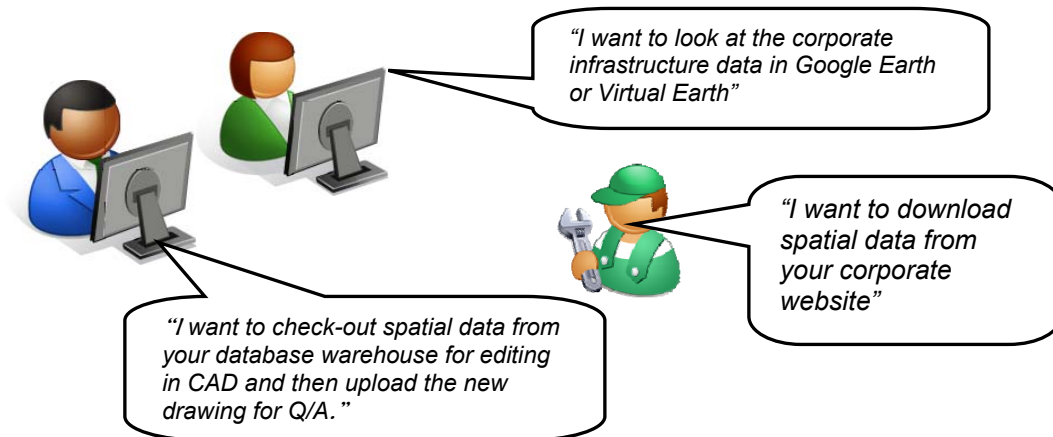


Defining User, Author and Administrator

When using a product with the range and scope of FME Server, it's to be expected that there will be a number of different roles available to different users.

User

- A person who accesses data via an FME Server service.
- No experience of FME required, indeed they need not even be aware of FME - desktop or server.



Author

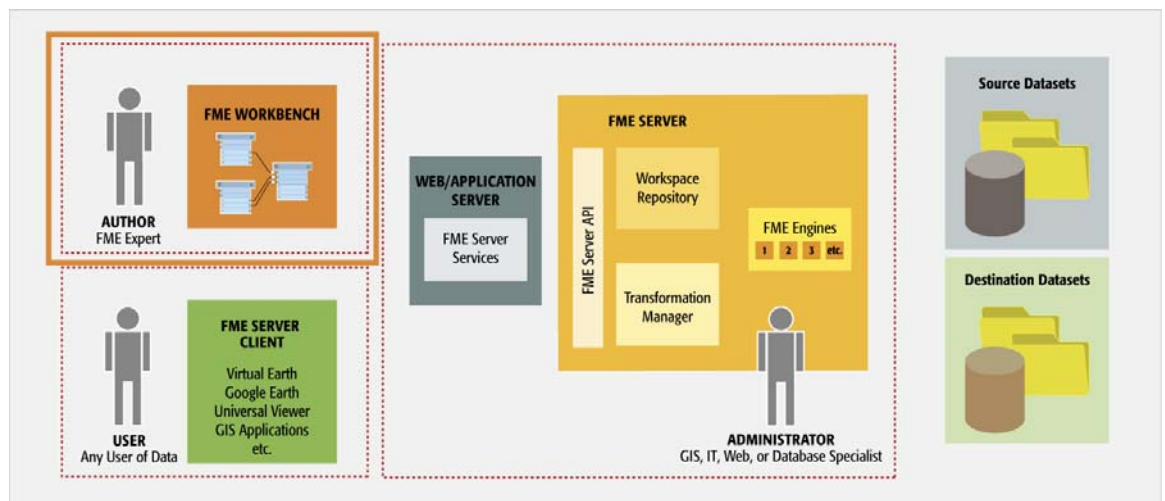
- Someone who creates workspaces and publishes them to the FME Server for use by users.
- Author is an experienced FME user with a good understanding of published parameters and the source data formats being used.

Administrator

- Person responsible for installing and maintaining FME Server and its related services. The administrator's tasks include:
 - Planning system architecture
 - Installation of prerequisites
 - Installation of FME Server
 - Setting up services
 - Monitoring services
 - Troubleshooting
 - Scaling (when required)

FME Server & FME Workbench – A Server Client Pair

- FME Workbench is the authoring environment for FME Server. It is part of the FME Desktop product and is not included as part of FME Server product.
- Workbench itself is a *client* of FME Server and has been enhanced to create, edit and publish workspaces specifically for the FME Server.
- Workspaces created in workbench can be published to the FME Server where they are held in a repository
- The repository database on the FME Server stores all of the relevant information about the workspace including published parameters, feature types, and source and destination datasets.
- Workspaces can be queried and run on the FME Server by FME Server clients and services



Above: User Roles Diagram – Showing Author

04. Downloading a Workspace from FME Server



Download workspaces from FME Server using Workbench

As a client of FME Server, Workbench can transfer workspaces to and from the Server workspace repository. Downloading a workspace allows a user to carry out local editing.

Example - Downloading a Workspace

Workspaces are downloaded using a wizard started by either File > Download from Server, or the download tool on the Workbench toolbar.

1) Start FME Workbench

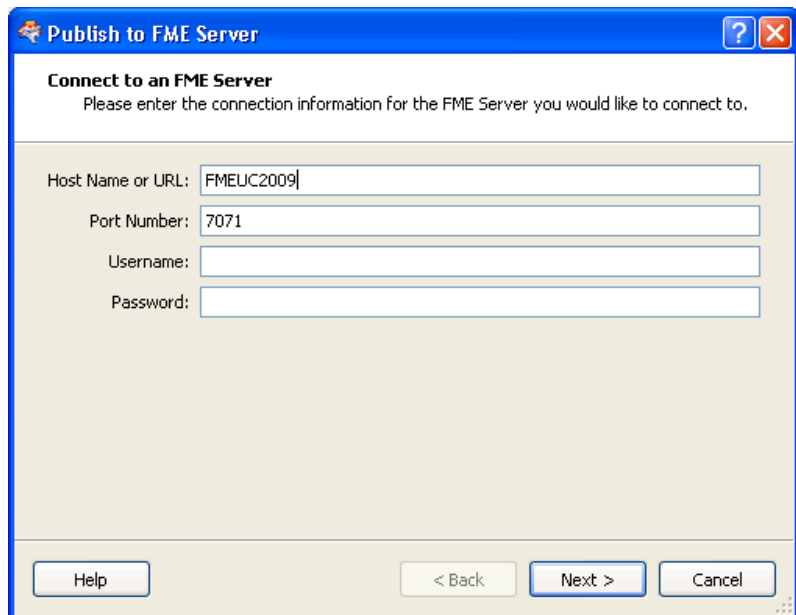
Start FME Workbench and create a blank workspace.

2) Start Download Wizard

From the *File* menu select *Download from Server...*

The first panel asks for the host and connection information. Enter your machine's host name and leave the other defaults as they are. Click the Next button.

Right: TCP/IP connection parameters to the FME Server in FME Workbench 2009




In FME 2009 Workbench connects to the FME Server using either TCP/IP or SOAP. If connecting via TCP/IP the host name of the FME Server and a port number are required to make a connection. If connecting via SOAP the URL of the SOAP service is required.

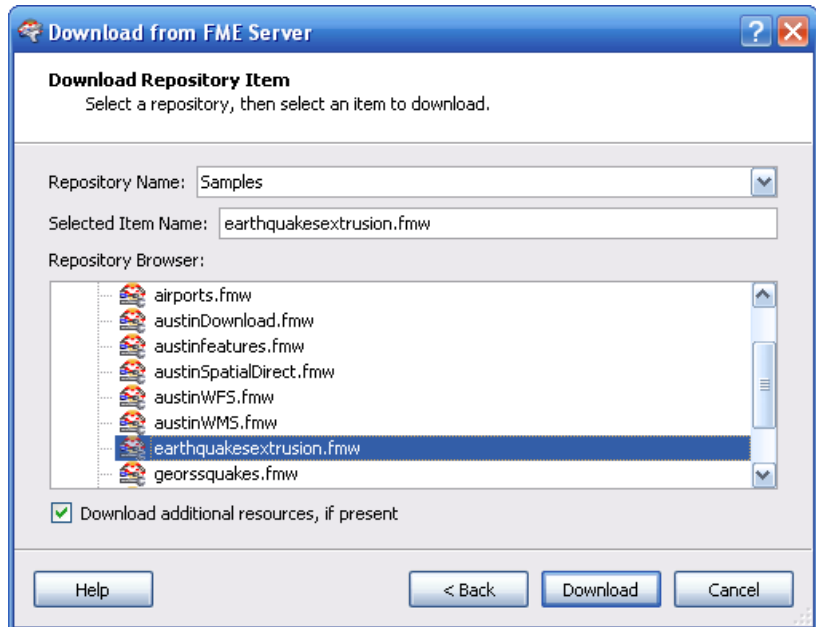
3) Choose the Repository and Workspace

Select the *Samples* repository and the choose workspace *earthquakesextrusion.fmw*.

Click the download button.

Remember, a **repository** is a folder on the FME Server that allows us to separate workspaces and other resources into individual containers.

Right: Select the Repository and then the workspace you want to download



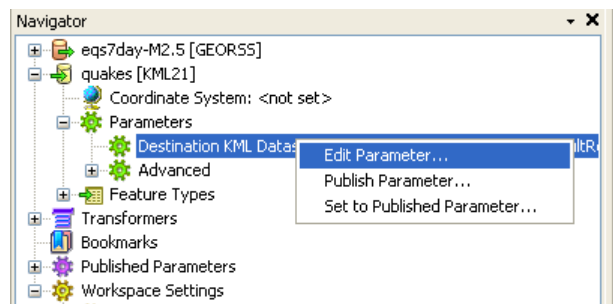
4) Save the Workspace

Save the workspace locally in the directory *C:\FMEData\Output\TrainingModule3*

5) Set Destination KML dataset

Set the destination KML dataset to be *C:\FMEData\Output\TrainingModule3\quakes.kml*

Right: Updating Destination KML Dataset



6) Run the Workspace

Run the workspace (using FME Workbench) and review the results within Google Earth. The file should be located in *C:\FMEData\Output\TrainingModule3\quakes.kml*.

As you can see multiple earthquakes have occurred within close vicinity and are currently intersecting one another. In the next section we will use published parameters to control the size and shape of the cylinders.

05. Published Parameters and FME Server



Any parameter published inside a workspace is available to a user in the FME Server

Published Parameters and FME Server

Published parameters in workspaces intended for the FME Server help in two ways:

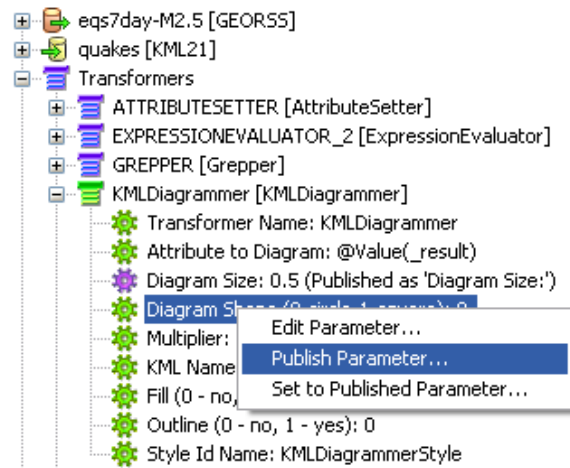
- Any client of FME Server can query the workspace and find out what parameters are available
- When FME Server runs a workspace, any published parameters can be set dynamically

This one of most powerful things about FME Server!

For example, we could publish the destination coordinate system parameter in Workbench. When a user runs this workspace using a web form the user will see the option to set the coordinate system and can select the coordinate system for destination data.

1) Publish a Parameter

The workspace uses a custom transformer called *KMLDiagrammer*. In the navigation pane, locate this transformer and publish the parameter *Diagram Size*. This parameter controls the diameter of the shape representing the earthquake.



Right: Publishing parameters

2) Publish another Parameter – *Diagram Shape (0-circle,1-square)*

In the navigation pane browse to the *Diagram Shape* parameter and publish it.

This parameter controls the shape used to represent the magnitude of earthquakes in the KML output.

3) Save the Workspace

Save the workspace in preparation for publishing back to the FME Server.

06. Publishing a Workspace to FME Server



As a client of FME Server, Workbench can be used as a means to publish workspaces

From Workbench we can publish workspaces to FME Server using *File > Publish to Server* on the menu bar, or the matching toolbar icon.

1) Start the Publish Wizard

Before publishing your workspace it is best practice to run it in Workbench to see if there are any errors. However, rather than using the default *Run* button we can view/modify the published parameters by using *Prompt and Run*. To do this use *File > Prompt and Run Translation*.

If everything ran through correctly – with the correct parameters published – choose *File > Publish to Server*.

2) Accept the Connection Parameters

Use the same connection parameters as when downloading the workspace.

Host: FMEUC2009
Port Number: 7071

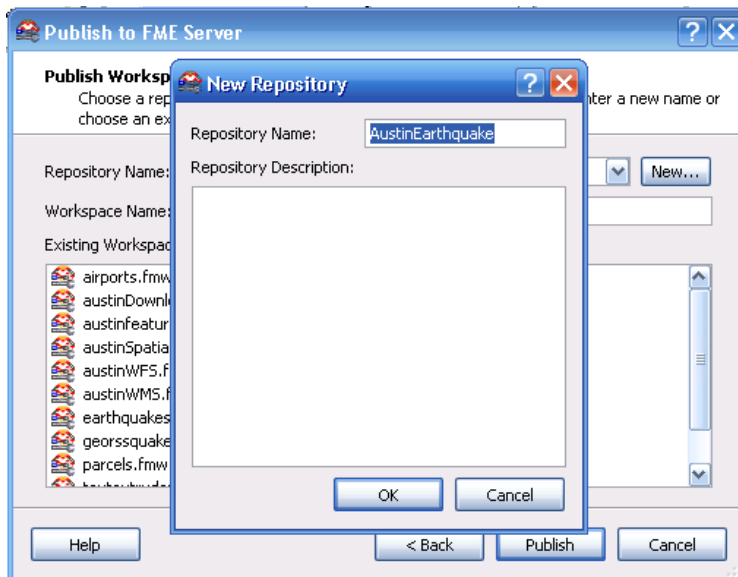
3) Create a New Repository

Importantly, each repository is separate from any other. This allows multiple workspaces of the same name – in a single FME Server instance – provided they are in different repositories.

By grouping workspaces we can ensure workspaces can only be viewed by the appropriate users.

Click the **New** button and enter *AustinEarthquake* to create a new repository.

Right: *Creating new repository*



4) Publish the Workspace

Click the *Next* button and then click *Next* on the following screen as we do not want to upload any data. Click the *Publish* button to upload the workspace.

5) Register the Workspace

When uploading a workspace the user is prompted to register the workspace with a service. In this case we will just register it with the Job Submitter service.

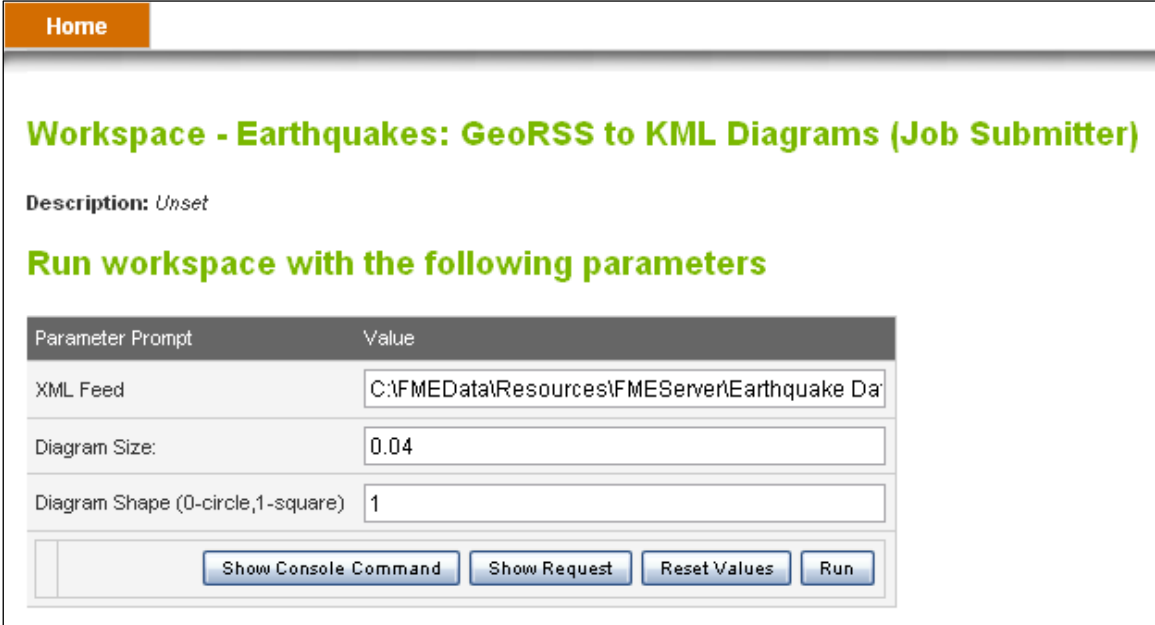
Check the FME Workbench log window to confirm the workspace was published successfully.

6) Run the Workspace in the FME Server

Enter the URL to your FME Server in a web browser: <http://<HostName>/fmeserver>

Navigate to:

*Job Submitter -> AustinEarthquake-> earthquakesextrusion.fmw -> **Configure***



Parameter Prompt	Value
XML Feed	C:\FMEData\Resources\FMEServer\Earthquake Da
Diagram Size:	0.04
Diagram Shape (0-circle,1-square)	1

Buttons: Show Console Command, Show Request, Reset Values, Run

Above: Running the translation from within the FME Server Web UI

Notice how the *Configure* page has dynamically fetched the published parameters in your workspace, including *Diagram Size:* and *Diagram Shape (0-circle,1-square)*.

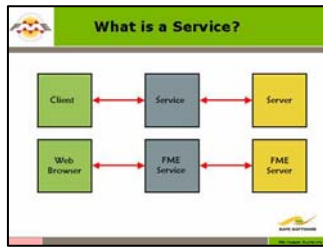
Set the diagram size to a value of **0.04** and the diagram shape to a value of **1**.

Run the workspace by clicking on the 'Run' button.

Once submitted the workspace is run and the resulting KML file is placed in the destination directory. Browse to the destination data set directory to ensure the data is there (should be: *C:\FMEData\Output\TrainingModule3\quakes.kml*).

Open the KML file in Google Earth and zoom to the data. Compared to the start of this exercise you should see a change in the size and shape of the symbols representing the earthquakes.

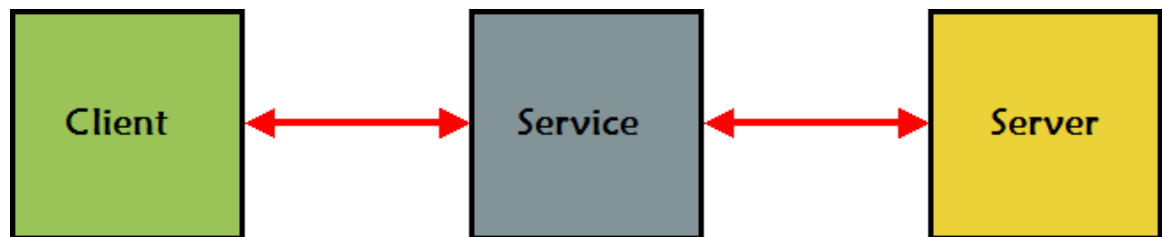
07. What is a Service?



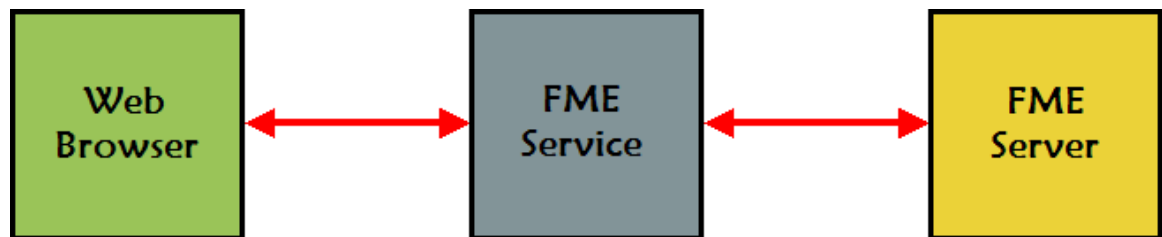
Many standard operations using the FME Server will take place using something called a 'Service'.

What is a Service?

In the simplest of terms, a service is a piece of software that handles communications between a client and a server. In other words it is a tool that allows a user to access complex functionality using a much simplified interface.



In terms of the FME Server, the client is often – but not always – a web browser which passes requests to the FME Server (actually the FME Server API) using a service.



Why use a Service?

A service allows us to send specific types of requests to the FME Server and to provide results to client application in a specific way.

For example, instead of just running a workspace, you can have a web page ask for the results of the workspace as a zipped up package of data.

FME Server Services

There are a number of services provided by default with FME Server, to carry out common tasks. However, users are also able to create their own service to carry out a specific task.

How to Use a Service

Up to now we've been using the Job Submitter service – which is a way of allowing the FME Server Web UI to instruct an FME Engine to run a translation.

But the Job Submitter is not the only service we have available.

Register	Service Name
<input type="checkbox"/>	Data Download
<input type="checkbox"/>	Data Streaming
<input type="checkbox"/>	Job Submitter
<input type="checkbox"/>	KML Network Link
<input type="checkbox"/>	OGC Web Feature Service
<input type="checkbox"/>	OGC Web Mapping Service
<input type="checkbox"/>	SpatialDirect

When a workspace is published to the FME Server the author is presented with the following dialog (*left*):

At this point the workspace is already published, and this is for registration of the workspace against different services.

Putting a check mark against one of these boxes registers the workspace as capable of being used by that service.

As we'll discover, not every workspace is capable of being used by every service.



It is **VERY IMPORTANT** to realize that a workspace can be registered against one service, many services, or even no services at all!

08. Data Download Service



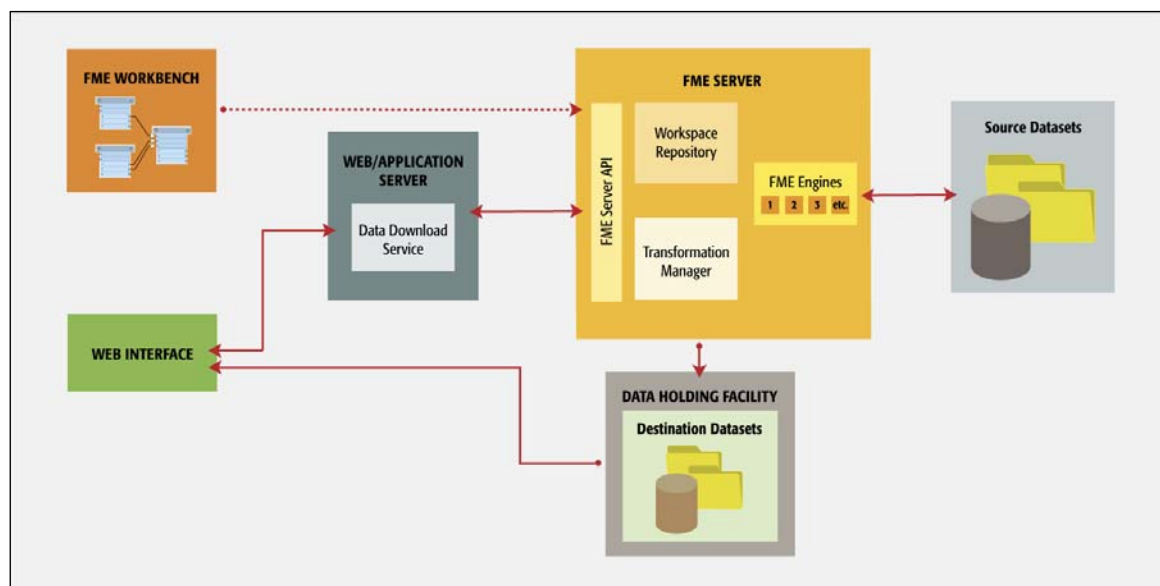
Data Download services are the ideal tool for allowing a user to quickly obtain spatial data in the format of their choice.

What is a Data Download Service?

A data download service is when – instead of writing output to the location defined in the workspace – a user wishes FME Server to store the output itself and make it available for them to retrieve and use locally.



In this scenario a user runs a translation and – instead of the output being written to a permanent destination dataset – the resulting data is written to a temporary data holding facility. The data download service provides a clickable link by which the user is able to download the data.



Above: Schematic Diagram of a Data Download Service

Example – Data Download

This example demonstrates the data download service and the power of integrating it with an existing web mapping client. The main advantage of this is that the user can easily select the area they wish to clip by drawing a polygon on the Google Maps pane.

1) View Template

A template web application has already been created. If you open the following URL in Firefox you will see the structure of the site:

http://fmeuc2009/ClientDemos/GoogleDemo/GoogleMaps_Demo.htm

We will now use FME Workbench to dynamically create the content for the web application.

2) Populate the Web Application

We are now going to use run a workbench which will populate the website. When run, the workspace does the following:

- Create the legend including icons
- Generate static KML which will then be overlaid on Google Maps
- Focus the centre point of the map on the data

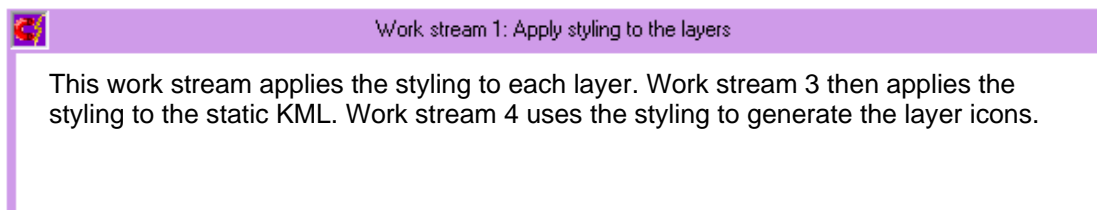
The web application has already been unzipped within the ROOT folder of Tomcat. Navigate to:

C:\Program Files\Apache Software Foundation\Tomcat 6.0\webapps\ROOT\ClientDemos\GoogleDemo

Open the *GoogleMapsDemo_Admin.fmw* workspace in Workbench (located in the directory above).

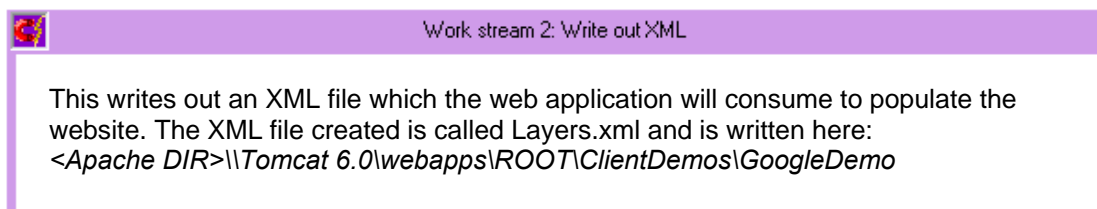
This workspace is doing a lot of tasks and as a result looks hugely complicated. What is important to note, is that this workspace is **NOT** translating spatial data. It is merely populating the web application.

The inputs to the workspace are the feature classes which you wish to view within the Google Maps application. Below is a brief description of what each element in the workspace is doing.




Work stream 1: Apply styling to the layers

This work stream applies the styling to each layer. Work stream 3 then applies the styling to the static KML. Work stream 4 uses the styling to generate the layer icons.




Work stream 2: Write out XML

This writes out an XML file which the web application will consume to populate the website. The XML file created is called Layers.xml and is written here:
<Apache DIR>\Tomcat 6.0\webapps\ROOT\ClientDemos\GoogleDemo

 Work stream 3: Generate Static KML

This generates static KML files, one for each feature type which will be rendered on the map.

 Work stream 4: Generate layer icons

This generates the symbol icons, each layer will have one symbol icon displayed to the user within Google Maps. They are written here:
<Apache DIR>\Tomcat 6.0\webapps\ROOT\ClientDemos\GoogleDemo\LayerIcons

The workspace has already been setup for you. We are going to load some transit data from Austin into the Google Maps application.

Inspect this data within FME Universal Viewer:

Format: Autodesk MapGuide Enterprise SDF
Dataset: C:\FMEData\Data\Transit\Transit.sdf

Within FME Workbench, click on the *Run Translation* button and then study the Layers.xml file and the icons created (tip, see purple boxes above for the locations of these).

The Google Maps web application should now have been populated with the Austin transit data.

In Firefox navigate to http://fmeuc2009/ClientDemos/GoogleDemo/GoogleMaps_Demo.htm

You will see that the legend has been created and the KML layers overlaid on Google Maps. The map has also centered over our data.

3) Data Download Workspace

Up until now we have just used FME to generate content for the website. If you were to try use the web application it would not work. This is because no workspace has been uploaded to FME Server, so any request made for data will fail. Once uploaded, the workspace will be run by FME Server every time a request for data comes in. We will now generate the data download workspace and upload it to FME Server.

Open the following workspace:

C:\Program Files\Apache Software Foundation\Tomcat
6.0\webapps\ROOT\ClientDemos\GoogleDemo\GoogleMapsDemo.fmw

Study the workspace. This workspace clips the source destination data; it then styles the data and writes out to a generic writer. The key things to note about the workspace are as follows:

- The Generic Writer allows the user to select their desired output format within the web application. This is extremely important as it gives the user control on how they want their data returned. Since FME has over 250 supported formats, the potential choice for the user is huge!!

- The clipper uses the geometry created within the Google Maps application when the user draws a polygon to clip the source datasets.
- Styling is applied to each layer. All the parameters associated with the styling are published and they are set dynamically within the Google Maps application.

4) Add Source Datasets

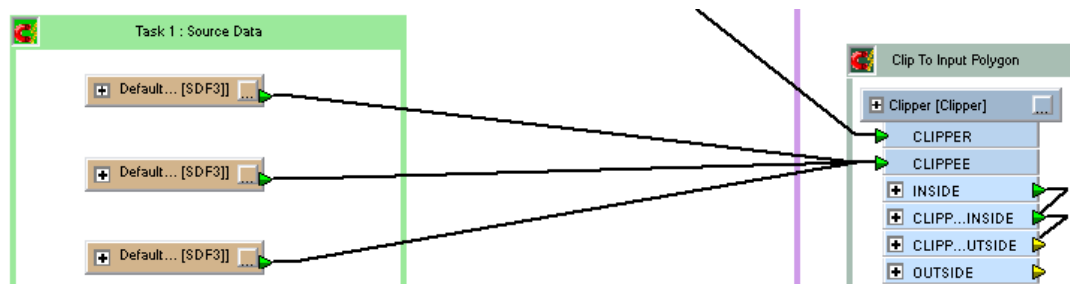
The first thing we need to do is add the source datasets to our workspace. It is this data which will be read when the user submits a request to FME Server. Add the Austin transit data we studied earlier.

Format: Autodesk MapGuide Enterprise SDF
Dataset: C:\FMEData\Data\Transit\Transit.sdf
Coordinate System: TX83-CF

When prompted add all three feature types. The web application supports a maximum of 10 layers.

5) Connect the Source Datasets

Connect all of the source datasets to the *CLIPPEE* port on the *Clipper* transformer.



Above: Connecting the source datasets up to the *Clipper* transformer

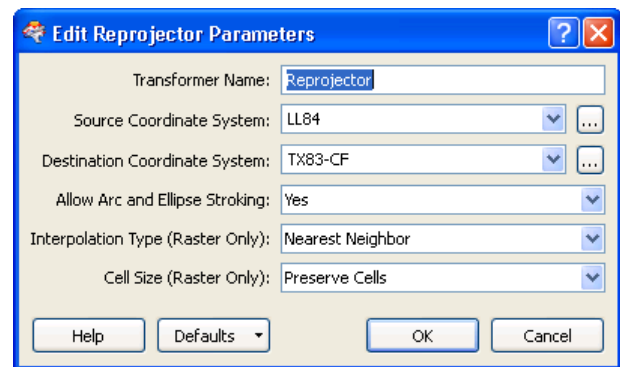
6) Set Coordinate System on Reprojector Transformer

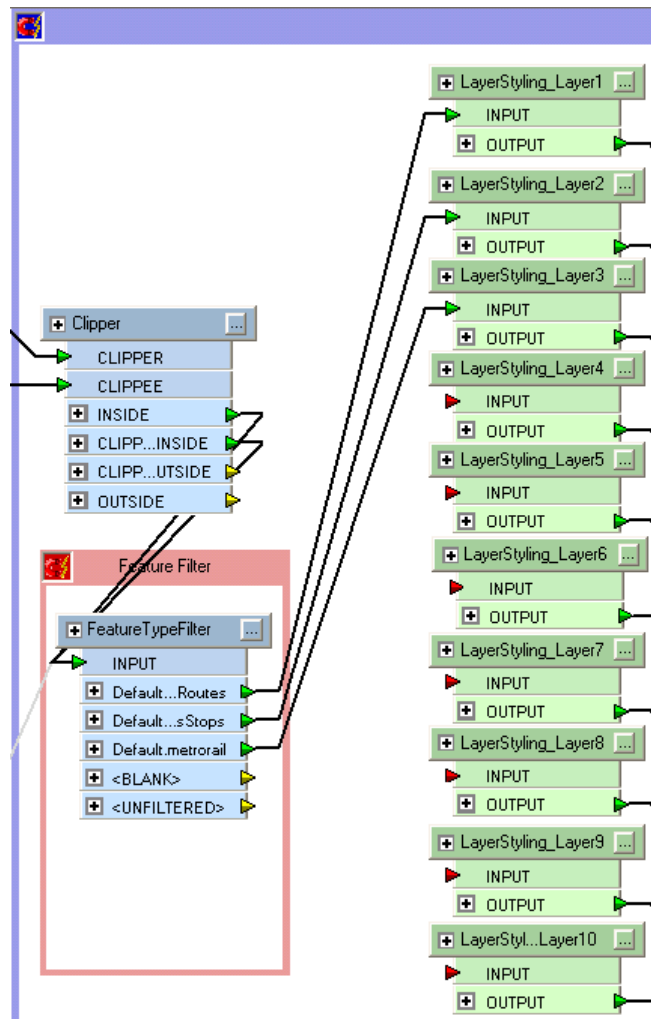
The 'Task 2: Clip To Input Polygon' bookmark in the top left corner of the workspace is responsible for clipping the source data. The creator transformer has a published parameter which takes in the geometry created by the user in Google Maps when they draw a polygon.

Since this geometry is in latitude/longitude and our data is in Texas State Plane, before we clip the data, we must reproject it.

Open the *Reprojector* transformer and ensure the *Destination Coordinate System* is set to *TX83-CF*. This value should always match the source dataset coordinate system.

Right: Setting the destination coordinate system so it matches the source dataset





7) Populate FeatureTypeToFilter Transformer

Currently our features have all been merged into 1 data stream because of the clipper. We now need to filter them out into individual feature classes again. Locate the *FeatureTypeToFilter* transformer in the top purple bookmark.

Open the transformer, click on the *Update* button and then *Sort*, you should see your layers populate the box.

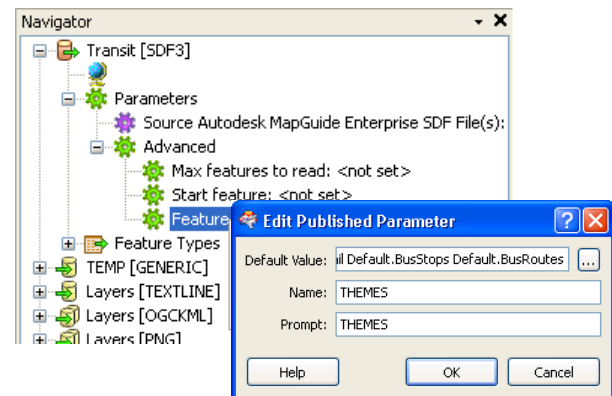
Close the transformer and link the feature types up to the *LayerStyling* transformers. See diagram to the left.

Left: *FeatureTypeFilter* connected up to the *LayerStyling* transformers.

8) Set Feature Types to Read parameter

Within the web application the user can select which layers they wish to be included in the clip. To make this work we need to ensure the *FeatureTypeToRead* parameter on the source dataset is published.

Locate the Transit source dataset and publish the *Features Types to Read* parameter, set the *Default Value* to all three layers and the *Name* and *Prompt* both to **THEMES** (case sensitive). It must be named **THEMES** as the JavaScript code within the Google Maps application relies on this naming convention.



Right: *Publishing a parameter*

9) Publish to FME Server

We are now ready to publish our workspace to FME Server. Publish using the following parameters.

Host Name	FMEUC2009
Repository	Create a new repository called <i>Demos</i>
Workspace Name	GoogleMapsDemo.fmw
Additional Resources	Leave default
Services Registered	Data Download

10) View workspace within FME Server

Enter the URL to your FME Server in a web browser: <http://<HostName>/fmeserver>

Navigate to:

Data Download -> Demos -> GoogleMapsDemo.fmw -> Configure

You can see all the published parameters here. Most of them relate to styling which will be auto-populated by a color-picker in the web application. In fact all these values will be populated by the user in the web application without them knowing it.

11) The Show Request Button

Within the configure page if you scroll down to the bottom you will see a *Show Request* button. Click on this button.

All requests sent to FME Server are ultimately a variation on a HTTP request. The “Show Request” button simply shows the syntax for the job request which is about to be run.

This is a very useful tool for building your own web applications that access FME Server services, because you can copy the HTTP request and embed it on your own web site or application. The Google Maps application builds a POST request up based upon selection users make in the interface.

The page shows two ways to send a request to an FME Server service:

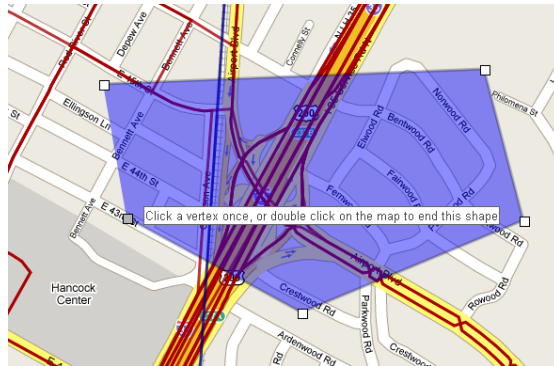
1. The URL method, also known as *GET HTTP request*, inserts all request parameters into a single URL string
2. The HTML form method, also known as a *POST HTTP request* sends all parameters as values in a web form when the form is submitted to the web server

12) Google Maps Application

The web application is now complete. In Firefox navigate to:

http://fmeuc2009/ClientDemos/GoogleDemo/GoogleMaps_Demo.htm.

Zoom in to a study area somewhere in the center of Austin. Then start to generate a request. Draw a polygon of a study area, select the layers you want to clip on the left. Chose the output format, if it is a format which supports styling you can also choose the styling for each layer by clicking on the *Styler* button.

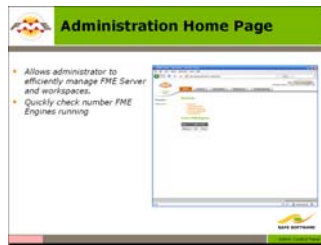


Left: Drawing Polygon

Before you send the request you can see the request which you have built up. Click on the dark green strip to the right of the legend. This shows us the POST and GET request that is going to be sent to FME Server. It uses the same structure as we saw when we clicked on the *Show Request* button.

Click on the *'Order Data'* button. The request is sent to FME Server where it is processed and the results returned back to the user as a Zip file.

09. Administration Control Panel



The administration page allows the administrator to efficiently manage FME Server and the workspaces stored on it.



- The **Services** tab allows the administrator to modify the properties of existing Services.
- The **Repositories** tab allows the administrator to create and delete repositories and publish workspaces, custom transformers and custom formats to repositories.
- The **Workspaces** tab allows the administrator to register/unregister workspaces against a service.
- The **Job Management** tab allows the administrator to view jobs which have been submitted to the FME Server.

Example – Managing Repositories and Workspaces

1) Enter URL

<http://<Trainer'sHostName>/fmeserver/services/fmejobsubmitter/Austin Earthquake>

2) Run Workspace

When the trainer tells you, hit Run. Within her/his Job Management page, jobs will begin to appear under queued/running and once the translations have run through, the history including a log file for each translation will also be visible.

3) Purge Temporary Data

The trainer will then demonstrate how to purge jobs from the Job History page.

10. Architectural Considerations



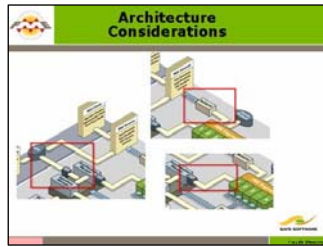
Further study of the individual components which make up FME Server

Components FME Server

- Services and other Clients use the FME Server API to send request to FME Server via HTTP.
- The SOAP service is another client that uses the FME Server API to send requests to FME Server via TCP/IP.
- FME Server Transformation Manager queues jobs
- FME Server Repository Manager holds information about workspaces
- FME Engines perform the translations and return results

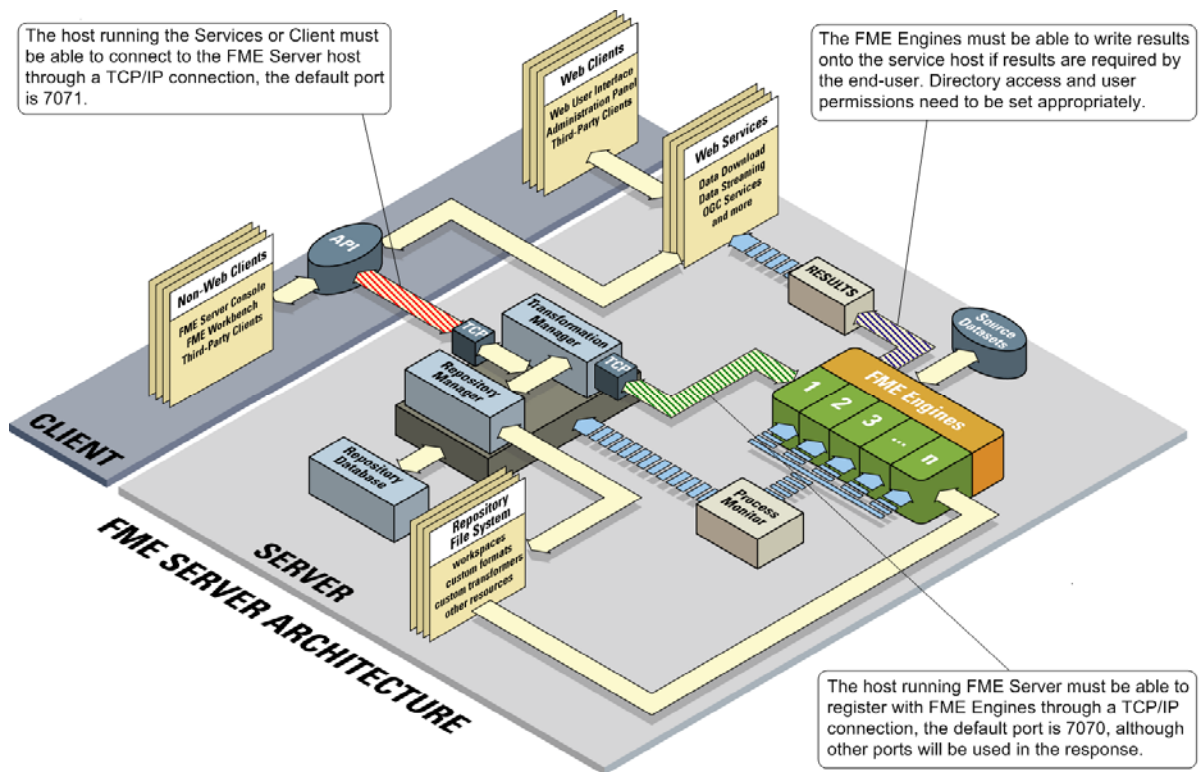
Usually these components are grouped into 3 groups:

- Services and Clients (all web applications provided with FME Server)
- FME Server (Transformation Manager and Repository Manager)
- FME Engine(s)



What needs to be considered in a distributed environment?

All three of these groups; *Services, FME Server, FME Engine(s)* can be run on separate physical or virtual hosts. They do not even need to be the same operating system! We won't cover this topic completely, but here are some considerations:



Individual Components in FME Server Installer

The FME Server installer allows us to install individual components on separate hosts to support configuration in a distribute environment. We won't try this but here is an example;

- We can install the FME Server without any Services or FME Engines.
- Then on another host we could install the Services only and by specifying the correct FME Server host in the installer the services would know where to direct their requests.
- Finally we could install only FME Engines on a third host and specify the FME Server host with which the FME Engines will register.

11. Data Streaming Example

Up until now we have only worked with the data download service. This service returns a web page with a link to the data. The data streaming service returns the data itself. In this section we will modify our workspace and publish it to the data streaming service on FME Server.

A good use case for this is when the URL for the data streaming service is posted into a client that actively downloads the contents on a regular basis – as a GeoRSS reader would – then you have a feed: something significantly different to a regular data download service.

What Formats Can be Streamed?

In theory, any workspace that returns a single output file can be used as a data streaming service; for example AutoCAD DWG format could be streamed, ESRI Shape format could not.

However, in practice data will only be streamed when there is a suitable client to read that feed. Some of the main formats output using the data streaming services include:

- RSS
- GeoRSS
- JSON
- GeoJSON
- KML

Example – Data Streaming

1) Open Data Download workspace

We only need to make some minor modifications to the workspace we used for the data download service.

Open the workspace located here:

```
C:\Program Files\Apache Software Foundation\Tomcat  
6.0\webapps\ROOT\ClientDemos\GoogleDemo\GoogleMapsDemo.fmw
```

2) Setting the MIME type

A MIME header is a component of a file (or email message) that is capable of indicating the content type of the file; for example *Content-Type: text/plain* indicates a simple text file.

Workspaces can store the MIME type of the output datasets, and transmit that information to the data streaming service.

What this means is when the data is returned to the client (e.g. Firefox), the browser will understand the content of the file. It can therefore assign the correct application to open the file (for example open a KML file in Google Earth) without being prompted by the user.

To set the MIME type we need to add a writer of that format to the workspace. Adding this writer ensures the content-type is incorporated into the HTTP which is sent back to the client.

Add *GIF Rasterizer* destination dataset to the Workbench. Use the following settings:

Destination Format	GIF Rasterizer
Destination Dataset	C:\TEMP\Output.gif

When prompted to whether you want to add a new feature type, select **No**.

NB: Be sure not to pick the format GIF (*Graphics Interchange Format*) – this would be incorrect!

3) Publish to FME Server

We are now ready to publish our workspace to FME Server. Publish using the following parameters.

Host Name	FMEUC2009
Repository	Demos
Workspace Name	GoogleMapsDemo_Streaming.fmw
Additional Resources	Leave default
Services Registered	Data Streaming

4) View workspace within FME Server

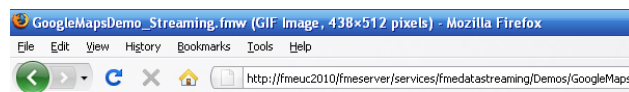
Enter the URL to your FME Server in a web browser: <http://<HostName>/fmeserver>

Navigate to:

Data Streaming -> **Demos** -> **GoogleMapsDemo_Streaming.fmw** -> **Configure**

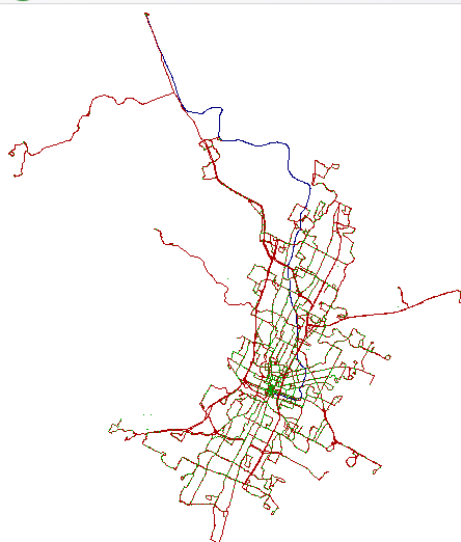
You will see all the published parameters associated with this workspace. Set the **FORMAT_GENERIC** parameter to **GIF**.

Click the Run button to run the translation.



A GIF image showing the entire Austin area (because we didn't provide any clipping geometry) will be returned.

Left: GIF image returned





www.safe.com

Safe Software values its customers' opinions very highly.
To provide feedback on FME training, access the feedback form at: www.safe.com/feedback.
For questions and concerns not covered by that form, please use the general feedback page at:
www.safe.com/company/contact/form.php.
...or email the Training Manager directly at: training@safe.com.