Hilltop Server

(Web Service)

20-May 2016

Contents

Hilltop Service

The server offers a set of native requests through the Service=Hilltop key. The use of proprietary protocols is often frowned upon, but it enables rapid delivery of features that aren't part of accepted standards. For example, the Hilltop service can prepare a graph for display on the client. Such activity is not standardised.

Site List

The SiteList request will deliver an XML document that shows the sites on offer. A typical example is:

```
<?xml version="1.0"?>
<HilltopServer>
<Agency>Hilltop Headquarters</Agency>
<Site Name="Alphabet River at Crossword"> </Site>
<Site Name="Anakina River at Duntroon"> </Site>
</HilltopServer>
```

Note the Agency element because you may need to put entries in the Hilltop Transfers table to convert the site names and measurements to your local settings.

You can ask the server to provide the location of the sites. Use the Location key and set this to either "Yes" or "LatLong". The Yes setting gives the easting and northing and the LatLong setting gives latitude and longitude in NZGD2000 format. For example:

```
http://host/data.hts?Service=Hilltop&Request=SiteList&Location=Yes
```

will return:

```
<?xml version="1.0"?>
<HilltopServer>
<Agency>Hilltop Headquarters</Agency>
<Site Name="Alphabet River at Crossword">
  <Easting>2561014</Easting>
  <Northing>5982791</Northing>
</Site>
<Site Name="Anakina River at Duntroon">
  <Easting>2129920</Easting>
  <Northing>5485470</Northing>
</Site>
</HilltopServer>
```

A URL to give latitude and longitude looks like:

```
http://host/data.hts?Service=Hilltop&Request=SiteList&Location=LatLong
```

and will return something like:

```
<?xml version="1.0"?>
<HilltopServer>
<Agency>Hilltop Headquarters</Agency>
<Site Name="Alphabet River at Crossword">
  <Latitude>-41.36010270</Latitude>
  <Longitude>173.60985234</Longitude>
</Site>
<Site Name="Anakina River at Duntroon">
  <Latitude>-45.73724556</Latitude>
  <Longitude>168.11528305</Longitude>
</Site>
</HilltopServer>
```

Filters

Other keys let you get site information to help build a map layer. The output is the same, but is filtered by the keys. You can provide a bounding box to the server and so only see sites inside the box. You can also give a measurement name, and the server will only show sites that have the given measurement. Example URL's are:

```
http://host/data.hts?Service=Hilltop&Request=SiteList&Location=Yes&
BBox= 2100000,5400000,2180000,5600000

http://host/data.hts?Service=Hilltop&Request=SiteList&Location=Yes&
Measurement=Flow
```

```
http://host/data.hts?Service=Hilltop&Request=SiteList&Location=Yes&
BBox= 2100000,5400000,2180000,5600000&Measurement=Flow [Water Level]
```

Bounding Box Filter

The BBox key accepts two easting and northing pairs by default. These define two points diagonally opposite to each other. The order of the pairs is not important, but the easting precedes the northing in each pair. The BBox key can be in lat/long if you wish and you use a shortened version of the OGC format. For example to send a bounding box in WGS84:

```
BBox=-46.48797124,167.65999182,-44.73293297,168.83236546,EPSG:4326
```

Valid EPSG numbers are:

EPSG:4326	WGS84 lat and long
EPSG:2193	NZTM 2000
EPSG:27200	NZMG
EPSG:4167	NZGD 2000

Life is simpler if you use eastings and northings in the same projection as the server is sending them. The server can convert from NZMG to NZTM2000, but not the other way.

Measurement Filter

You can also give a measurement name, and the server will only show sites that have the given measurement. The measurement can include a data source name if required. Note that you must use the names that the server will understand, not the names of measurements on your own system. This isn’t a problem with bog-standard surface water measurements like flow and rainfall, but it is important for Water Quality data. The simplest way to find out the measurement names in use at the server is to ask it for a measurement list.

Site List via a Collection

The site list request can accept a collection name, and will list the site names in the given collection. This feature is handy for a web site that wants to load a tree control or combo-box with a list of suitable sites. The web site can be crafted to always load a known collection name, but what appears is controllable through the collection. This means you don’t have to change the web site to offer extra sites. An example is:

```
http://host/data.hts?Service=Hilltop&Request=SiteList&Collection=WQSites
```

The Location and BBox qualifiers are still available if you want to generate a map layer. The Measurement qualifier is not available.

The server can build a list of sites based on all the collections in the project. This is useful when you have a set of collections based geography or measurements, but want to load a combo-box on a web site with all available sites. Give a collection name of "~All" and the collection name is case sensitive. The server will read all the collections in the project and build a new collection on the fly. It will remove duplicates. An example is:

```
http://host/data.hts?Service=Hilltop&Request=SiteList&Collection=~All
```

Site Parameters

The Sites table in the Hilltop database has columns in it that may be useful in a site list. The location, catchment area and altitude are stored in their own columns and all other parameters are stored in an XML document. The XML document contains the parameters that Lawa uses for the Monitoring Site Reference Data map layer plus any key-value pairs you created in the Extras tab of the Sites Editor. You can ask for selected site parameters with the SiteParameters keyword and give a list of the parameter names separated by a commas. For example:

```
http://host/data.hts?Service=Hilltop&Request=SiteList&
SiteParameters=CatchmentArea,Altitude,AirTown
```

You can also have "Location" as a parameter and the server will show the easting and northing. This is the equivalent of the "Location=Yes" key-value pair. Consult the Lawa documentation for the names of the columns used in the Monitoring Site Reference Data map layer, or look at the FieldInfo column in the Sites table with SQL Server Management Studio.

Measurement List

The Hilltop Server can tell you what measurements are available at a site, and will give the time range of available data. The measurement list includes all the time series types that Hilltop has, so the list will include Check Data and Quality Data. The Request key is set to MeasurementList and there must be a Site key set to the site name you require the information on. For example:

```
http://host/data.hts?Service=Hilltop&Request=MeasurementList&Site=Alphabet
    River at Crossword
```

The output is not shown here because it does go on a bit. Sufficient to say that the elements in the XML are hopefully obvious and contain all the important meta-data that is stored in the server's database.

Measurement List via Collection

This variation offers a short list of the site/measurement combinations in the given collection and gives the time range of the available data. The command accepts matrix collections and will only show a site/measurement combination if there is data in the file. Name the collection instead of a site name, for example:

```
http://host/data.hts?Service=Hilltop&Request=MeasurementList&Collection=WQSites
```

You can use the Site and Collection keywords together and the server will only show the measurements at the given site. This request is useful if the collection is a matrix collection with only measurements, or has measurements and site names. Note that the given site name must be in the collection if the collection has site names. In other words, this command is best used with a collection that just has measurements in it.

The server can build a list of measurements based on all the collections in the project. This is useful when you have a set of collections based geography or measurements, but want to load a combo-box on a web site will all available sites. Give a collection name of "~All" and the collection name is case sensitive. The server will read all the collections in the project and build a new collection on the fly. It will remove duplicates. An example is:

```
http://host/data.hts?Service=Hilltop&Request=MeasurementList&Collection=~All
```

You can still use the Site keyword with the ~All collection. This will just show the measurements at the given site.

Measurement List for all Sites

The request will work without giving a site name and in this case it will give a list of all the measurements in the data file. The list is not sorted and a measurement is listed in the order it is found in the data file. This command is good for loading a combo-box on a web page that shows the available measurements. A user can then select a measurement and see a map layer of sites that hve data for the measurement. An example is:

```
http://host/data.hts?Service=Hilltop&Request=MeasurementList
```

A typical reply is as follows. The data source name will be enclosed in brackets if it is needed.

```
<?xml version="1.0"?>
<HilltopServer>
<Agency>Hilltop Headquarters</Agency>
<Measurement Name="Stage" />
<Measurement Name="Flow" />
</HilltopServer>
```

You can ask for the units of each measurement. An example request is:

```
http://host/data.hts?Service=Hilltop&Request=MeasurementList&Units=Yes
```

A typical reply is:

```
<?xml version="1.0"?>
<HilltopServer>
<Agency>Hilltop Headquarters</Agency>
<Measurement Name="Stage"><Units>mm</Units></Measurement>
<Measurement Name="Flow"><Units>m3/sec</Units></Measurement>
</HilltopServer>
```

Collection List

The client can ask for a list of the collections and the items in it. For example:

```
http://host/data.hts?Service=Hilltop&Request=CollectionList
```

The server will reply with the entire project file in an XML document. Hilltop projects are usually in the “.hpr” format, but Hilltop Manager and Hydro will accept projects in XML. The XML sent by the server is the same as used by Hilltop programs.

An example of a project with a single matrix collection in it is:

```
<?xml version="1.0"?>
<HilltopProject>
<Collection Name="Flows">
  <Item><Measurement>Flow</Measurement></Item>
  <Item><SiteName>Anakina River at Duntroon</SiteName></Item>
  <Item><SiteName>Alphabet River at Crossword</SiteName></Item>
  <Item>
    <SiteName>Casey Farm</SiteName>
    <Measurement>Flow [Water Level for Small Streams]</Measurement>
  </Item>
</Collection>
</HilltopProject>
```

Data

The server can provide data with the GetData request and you must provide Site and Measurement keys to indicate which time series you require. You can also provide a tsType key to request data other than the StdSeries data. Examples are:

```
http://host/data.hts?Service=Hilltop&Request=GetData&Site=Alphabet
    River at Crossword&Measurement=Stage&From=1/1/2012&To=1/11/2012

http://host/data.hts?Service=Hilltop&Request=GetData&Site=Alphabet
    River at Crossword&Measurement=Stage&From=1/1/2012&To=1/11/2012
    tsType=StdQualSeries
```

The first URL asks for the standard series data, and the second URL asks for the quality series. The time range is given in the From and To keys, and the values can be in any format that Hilltop understands. The server will reply in ISO8601, but you don’t have to speak to it in this format.

A fragment of a reply is:

```
<?xml version="1.0"?>
<Hilltop>
<Agency>Hilltop Headquarters</Agency>
<Measurement SiteName="Alphabet River at Crossword">
<DataSource Name="Water Level" NumItems="1">
  <TSType>StdSeries</TSType>
  <DataType>SimpleTimeSeries</DataType
  <Interpolation>Instant</Interpolation>
  <ItemInfo ItemNumber="1"> <ItemName>Stage</ItemName>
    <ItemFormat>I</ItemFormat> <Divisor>1</Divisor>
    <Units>mm</Units>
    <Format>####</Format>
  </ItemInfo>
</DataSource>
<Data NumItems="1" DateFormat="Calendar">
<E><T>1960-01-01T01:00:00</T><I1>23826</I1></E>
<E><T>1960-01-01T09:00:00</T><I1>23516</I1></E>
<E><T>1960-01-01T18:00:00</T><I1>23208</I1></E>
</Data>
</Hilltop>
```

The DataSource node contains the meta-data and is a repeat of what was sent in the MeasurementList request. The Data node contains elements for each time stamp. The <E> node surrounds each time series element, the <T> element has the time, and the <I1> element has the value of each item.

Time Range

The From and To keys set the time range for the data retrieval. You can provide both keys or use the defaults by not providing a key. The From key defaults to the start of the data and the To key defaults to the end of the data. The values for From and To are any date and time formats the Hilltop normally accepts.

The server also accepts a time range given in the ISO8601 Time Interval format. This is a start time and finish time, separated by a "/" character. A simple example is:

```
    TimeInterval=2010-01-01T12:00:00/2011-07-01T14:00:00
```

ISO8601 also has the notion of a duration, being a length of time, and so there are four ways to construct a time interval. These are:

<start time>/<end time>	Both start and finish times are given
<start time>/<duration>	Read from the given start time for the given duration
<duration>/<finish time>	Set the start time back from the given end time
<duration>	Read the across the given duration

A duration in ISO8601 has the letter 'P' then one or more pairs of numbers and letters. The letters refer to the units, and are obvious choices. The letter 'M' is used for both months and minutes so the duration must have the letter 'T' in it before the time of day units are used. Examples are:

P1Y	One year
P10Y	Ten years
P3M	Three months
P3D	Three days
P1Y3M	One year and three months
PT3H	Three hours
PT10M	Ten minutes
PT30S	Thirty seconds
P3MT30M	Three months and 30 minutes, because you might need to.

Examples of a time interval that has a duration are:

```
    TimeInterval=2010-01-01T00:00:00/P1Y
    TimeInterval=P1Y/2011-01-01T00:00
```

The first example gives a year of data from the given start time, and the second example gives the year before the given finish time.

No Time Range

Not giving a time range is a request to give the most recent value. This was decided by OGC for the SOS standard and is a simple, efficient way of getting the most recent value. It works well for instantaneous data like stage, flow or temperature, but it is not sufficient for rainfall data because it always needs an interval. Thus a sample request to get the preceding 3 hours of rainfall before the end of the data is:

```
http://host/data.hts?Service=Hilltop&Request=GetData&Site=Alphabet
    River at Crossword&Measurement=Rainfall&Method=Total&Interval=3 hours
```

Relative Time Range

A relative time range is where you provide a single duration in the TimeInterval keyword and expect to read data across a time range. For example, "TimeInterval=P1M" is an attempt to get a month's data, but it's not obvious what the actual time range is because the finish time is not given. OGC have banned this construction in a SOS request but Hilltop assumes that the requested finish time is the end of the data. Thus "TimeInterval=P1M" sets a start time one back from the end of the data, and a finish time at the end of the data.

This behaviour may not suit reporting software that is trying to get near real-time data. Hilltop Server extends ISO8601 by allowing the "now" reserved word to be given as a finish time. An example to read three hours back from the time of the request is:

```
    TimeInterval=P3H/now
```

The word "now" is case sensitive.

Aligning the Start Time

Setting a time range back from the end of data or the current time is of little use for rainfall data because people are used to the totals being aligned to the start of a day, or the start of a month. The same comment applies if you are asking for statistics from the server such as the monthly average. Hilltop Server has an Alignment keyword that lets you set the start time on a boundary. Examples are:

```
Alignment=00:00             Align to the start of a day
Alignment=09:00             Align to 9 a.m.
Alignment=1 month           Align to the start of the month
```

The start time is moved forward from the given time, and you need to take account of this when crafting a query with a relative time range. For example, the server receives a request:

```
TimeInterval=P1M/now&Alignment=1 month
```

The server sets the finish time to be the current time of the request. It next sets the start time to be one month back from the finish time, and then moves the start time forward to the start of the next month. The nett affect is that this time range represents the current month. You will need to set a duration of P2M if you want data from the month before the current month.

Data from a Collection

You can ask the server to give data for all the items in a collection. The collection can be a matrix collection and the server will expand it. Use the Collection key instead of the Site and Measurement keys. For example:

```
http://host/data.hts?Service=Hilltop&Request=GetData&Collection=Flows
                   &From=1/1/2012&To=1/11/2012
```

Statistics

The Method and Interval and Alignment keys instruct the server to compute statistics of the data. The Method key determines the type of statistic and valid settings are:

Interpolate	Step through the data at the interval and give a value at each step
Average	Compute the average across the interval
Total	Compute the total in the interval
Moving Average	Compute the Moving Average at the interval, and shift it back half the interval.
EP	Exceedance Percentile
Extrema	Min, Mean, Max and amount of missing record

The results are aligned to the requested start time, and this means you must should set the start time if you want the values aligned to a particular time or use the Alignment keyword described earlier.

The Interval key accepts the interval for the statistic, and the format is the same as when giving an interval to Hilltop Hydro. Examples of the Method and Interval keys are:

```
http://host/data.hts?Service=Hilltop&Request=GetData&Site=Alphabet River at
   Crossword&Measurement=Flow&Method=Average&Interval=1 day

http://host/data.hts?Service=Hilltop&Request=GetData&Site=Alphabet River at
   Crossword&Measurement=Flow&Method=Moving Average&Interval=1 day

http://host/data.hts?Service=Hilltop&Request=GetData&Site=Alphabet River at
   Crossword&Measurement=Rainfall&Method=Total&Interval=6 hours
```

The EP method reads all the data to compute the probability density function and then expresses the data in the requested time range as an exceedance percentile.

The Extrema method finds the minimum and maximum value across and interval, and also gives the average value across the interval, and the amount of missing record in the interval. The missing record is in seconds. The results are sent as four item data, and the order of the items is min, mean, max and missing record. An Interval keyword is required with the Extrema statistic.

The timestamp associated with the average, total, and extrema statistics is at the end of the interval. This can make it hard for client software to display the results because monthly or annual averages appear to be in the next interval. For example, the average of all the data in 2015 is shown at zero hundred hours on 1-January-2016. The DateOnly=Yes qualifier will do Tideda's trick of making

00:00:00 be 24:00:00 on the previous day. The server suppresses the time of day, and just gives a date which is now in the year in which the averages applies to. The server also uses a different tag name to show you it has written the date instead of a date and time. Examples of time and date are:

```
<T>2016-01-01T00:00:00</T>
<D>2015-12-31</D>
```

The server sends a timestamp and value at the start time of the requested analysis when doing a fixed average or extrema calculation. This is done so that graphing software knows where to start drawing the graph from. Such software is only interested in the time, not the value, but the convention is that the value is the interpolated value at the start time. Client software that is just trying to load a table of the results should discard the first timestamp/value pair that it reads.

Rainfall Totals

Rainfall data is always totalled across an interval, and getting the server to provide the correct result requires particular combinations of the keywords. The simplest query is when the client software calculates the time range itself. For example, the relevant keywords to get three monthly rainfall totals are:

```
Method=Total&TimeInterval=2015-01-01/2015-04-01&Interval=1 month
```

An implied time range requires more care. The Alignment keyword helps, but you must remember that the Alignment keyword moves the start time forward. Thus to get the rainfall total in the previous three complete months, you need to set the duration to four months. For example:

```
Method=Total&TimeInterval=P4M/now&Alignment=1 month&Interval=1 month
```

This keyword combination will not return the total in the current month. This is because the interval is set to one month, and there isn’t a complete month of data in the final month. This behaviour is fine if you are trying to prepare a graph of recent rainfall and plotting the current month would be confusing to people who don’t look carefully.

Use the ShowFinal keyword to ask the server to give the total in the final, incomplete month. Thus the relevant keywords to show the three previous months, plus the rainfall this month is:

```
Method=Total&TimeInterval=P4M/now&Alignment=1 month&Interval=1 month&ShowFinal=Yes
```

All this comes back to how you would ask for the rainfall this month. The example above will work will work with the time interval set back to one, but the server will accept a short-hand request where you don’t give an interval. In this case, it takes the absence of an interval as a request to total across the time range. Thus the following keyword combinations do the same thing and will return the rainfall this month.

```
Method=Total&TimeInterval=P1M/now&Alignment=1 month&Interval=1 month&ShowFinal=Yes
Method=Total&TimeInterval=P1M/now&Alignment=1 month
```

A New Measurement Name

The statistics functions take a time series and turn it into another one. The receiving software may wish to store the derived data and wants to see it with a different measurement name. This is done with the SendAs keyword.

An example to fetch the last seven days of daily mean flow and send it as Daily Mean Flow is:

```
http://host/data.hts?Service=Hilltop&Request=GetData&Site=Alphabet River at
    Crossword&Measurement=Flow&Method=Average&Interval=1 day&TimeInterval=P7D&
    SendAs=Daily Mean Flow
```

The server does not check the realisation value against its database and it merely inserts the value into the XML. You can have a data source name enclosed in square brackets and this will go into the Data Source element in Hilltop XML and the procedure element in WaterML 2. Exercise care with the SendAs keyword when fetching data with a collection. The server applies the new measurement blindly and a mixture of measurements in your collection will all be sent as the same new measurement name.

Translating Client Specific Names

The client software may have its own set of names for sites and measurements and wants to use these instead of using your names. The LAWA web site in an example of this where it may ask for a measurement that is not the same name that you use. Provided you know in advance what the client is going to ask for, the client can have an Agency qualifier on their URL and then you can use this to set the correct site and/or measurement using the Hilltop Transfers Table. Consider the following example where both the flow and measurement are not your names:

```
http://host/data.hts?Service=Hilltop&Request=GetData&Site=Alphabet&
    Measurement=Stream Flow&Agency=LAWA
```

You need to create a LAWA agency and then set up transfer table entries. The Transfers system can translate site/measurement pairs, site names and measurement names. You decide which group to do, but in most cases you will create a transfers entry for each site name, and another for each measurement. Thus in the example above, you would select the Measurements group in the Transfers Editor and add a new identifier called "Stream Flow". Set the Hilltop Measurement to your measurement, typically "Flow [Water Level".

The Site/Measurement pair transfer group is checked first. This lets you override the transfer rules for a particular site and measurement. For example, for the site that LAWA calls "Alphabet" you wish the Stream Flow to translate to "Flow [Water Level for Small Streams]". You need to select the Site Measurement Pairs group in the Transfer Editor and then add a new identifier that has the site name enclosed in braces then the measurement. For the example above, the identifier is "{Alphabet} Stream Flow". Select the Hilltop site name and measurement from the combo-boxes in the editor.

Response Format

The response to a GetData request is always in XML but you can control the dialect with the Format keyword. The default is Hilltop XML with the time in ISO8601 format and the data in a separate element from the time. The data is sent as single item data and the server will apply the divisor from the Measurements table.

Hilltop Clients will override this setting because they want to see the data as is, and also want the data without any divisors applied. Native format is also useful if you want to see all the items in multi-item data presented together. For example, to see all the Gauging Results data, ask for any item, but also ask for the native format.

```
http://host/data.hts?Service=Hilltop&Request=GetData&Site=Alphabet
    River at Crossword&Measurement=Stage [Gauging Results]&Format=Native
```

The native format is the original Hilltop XML format. The time is in Mowsecs, and the data is in the same element as the time. Mowsecs is the number of seconds since 1-Jan-1940, and you can convert this to more common representations as follows. Of course, you knew these magic numbers already.

```
    Unix time = mowsecs – 946771200
    Excel time = mowsecs/86400.0 + 14611.0
```

The server can send WaterML 2 and this lets you use the statistics functions on the server but still have an OGC compliant response. Set the Format keyword to WML2, for example:

```
http://host/data.hts?Service=Hilltop&Request=GetData&Site=Alphabet
    River at Crossword&Measurement=Flow&TimeInterval=P7D&Format=WML2
```

Data Tables

The server can send data in tabular format ready for parsing by the .Net Data Objects. The response is an XML document with two tables in it and these contain information about the measurements and the actual data. There is also an in-line schema to guide the data set parser. A simple example is shown below, and the schema is removed for clarity.

```
    <?xml version="1.0" ?>
    <HilltopServer>
    <Agency>Hilltop Headquarters</Agency>

    <Measurements>
      <ColumnName>M1</ColumnName>
      <Measurement>Total Nitrogen</Measurement>
      <Units>g/m3</Units>
    </Measurements>
    <Measurements>
      <ColumnName>M2</ColumnName>
      <Measurement>Ammoniacal Nitrogen</Measurement>
      <Units>g/m3</Units>
```

```
</Measurements>

<Results>
  <SiteName>Alphabet River at Crossword</SiteName>
  <AsAt>2013-01-01T14:00:00+13:00</AsAt>
  <M1>42.5</M1>
  <M2>23</M2>
</Results>
<Results>
  <SiteName>Akina River at Duntroon</SiteName>
  <AsAt>2012-06-01T14:00:00+12:00</AsAt>
  <M1>2.5</M1>
  <M2>3</M2>
</Results>

</HilltopServer>
```

The Results table has the (umm) results in it. Each row has a site name and a timestamp. The remaining columns in the table have the value of a measurement. These columns are given arbitrary names, M1, M2, etc. This is done because the server can’t use a measurement name in case it has a blank in it.

The Measurements table provides the measurement and its units. The table also gives the column name of the measurement in the Results table. This lets you connect the measurement information to the column in the Results table.

The in-line schema defines the layout of both the Measurements and the Results table. It also prepares a caption for each measurement column in the Results table. The caption has the measurement name plus the units in parentheses.

The C# example below is the code behind for a web page that displays the results table in a data grid. It works by asking the server for the XML document, and then loads the XML into a DataSet object. The DataSet holds the tables, and you can then get the DataGrid object to show the results table. The code in the middle of the example reads the captions from the columns in the table and sets the heading in the data grid.

```
protected void Page_Load(object sender, EventArgs e)
  {
  String url = "http://host/data.hts?Service=Hilltop&Request=RecentDataTable&
                    Collection=WQ Table";
  DataSet ds = new DataSet();
  HttpWebRequest request = (HttpWebRequest) WebRequest.Create(url);
  HttpWebResponse response = (HttpWebResponse)request.GetResponse();
  ds.ReadXml(response.GetResponseStream());

  if (ds.Tables["Results"] == null)
    {
    // Something went wrong.  Look at the response again and panic.
    // Insert thoughtful code here……
    response.Close();
    return;
    }

response.Close();  //House rules say we must close the response object

// Get the data grid on the job. Build the columns manually so we can set the
// header text from the caption in the Results table

  DataGrid g = new DataGrid();
  g.AutoGenerateColumns = false;
  foreach (DataColumn col in ds.Tables["Results"].Columns)
    {
    BoundColumn bc = new BoundColumn();
    bc.DataField = col.ColumnName;
    bc.HeaderText = col.Caption;
    g.Columns.Add(bc);
    }

  g.DataSource = ds.Tables["Results"];
  g.DataBind();
  form1.Controls.Add(g);
  }
}
```

Recent Data in a Table

A lead-in web page may show a table of the most recent reading for a set sites and measurements. This is especially handy for water quality sites where each row is a site name, and each column is a measurement. The server can prepare this table using a collection. An example is:

```
http://host/data.hts?Service=Hilltop&Request=RecentDataTable&Collection=WQ Table
```

The server will reply with an XML document that has been crafted for parsing by the .Net Data objects, as described above.

Dual Collections

The command will accept the names of two collections to provide more choices than a Chinese banquet. The scenario here is that a web page has two combo-boxes on it, one offering a list of sites, and the other a set of measurements. The site combo will have entries for various regions or zones, and the measurement combo will offers choices of measurements such as Nutrients, Faecal's etc. The entries in both combo-boxes are the names of collections. The ever alert user will make their choice and the server will provide the data. An example is:

```
http://host/data.hts?Service=Hilltop&Request=RecentDataTable&SColl=Northern&
   MColl=Nutrients
```

The names SColl and MColl hint at being a site collection and a measurement collection, but the software makes no assumptions. It just copies both collections into a third collection and then expands this against the selected data file. It does not check for any duplications that result from this process.

Site Parameters

As mentioned for the SiteList request, the Sites table in the Hilltop database has columns in it that may be useful in the table. The location, catchment area and altitude are stored in their own columns and all other parameters are stored in an XML document. The XML document contains the parameters that Lawa uses for the Monitoring Site Reference Data map layer plus any key-value pairs you created in the Extras tab of the Sites Editor. You can ask for selected site parameters with the SiteParameters keyword and give a list of the parameter names separated by a commas. For example:

```
    http://host/data.hts?Service=Hilltop&Request=SiteList&
    SiteParameters=Location,CatchmentArea,Altitude,AirTown
```

"Location" can be a site parameter and the server will show the easting and northing. Consult the Lawa documentation for the names of the columns used in the Monitoring Site Reference Data map layer, or look at the FieldInfo column in the Sites table with SQL Server Management Studio.

All Data in a Table

This command runs Hilltop Prep and returns the data in an XML document. You name a collection, and give a time range. Prep is run with the Group By Site option set and this means that the data is listed in site order. An example is:

```
http://host/data.hts?Service=Hilltop&Request=DataTable&Collection=WQSites&
    From=1/1/13&To=1/7/13
```

You can also ask the server to build a collection on the fly. You provide a list of sites and measurements, and the server will build a matrix collection and then expand it. It doesn't matter if a site or measurement isn't in the data file. The expanded collection won't ask for data that isn't there. Site names and measurements are separated by commas. Enclose a site name or measurement in double-quotes if the name has a comma in it. An example is:

```
http://host/data.hts?Service=Hilltop&Request=DataTable&
Site=Anakina River at Fairway,Anakina River at Duntroon&
Measurement=pH,Conductivity&
From=1/1/13&To=1/7/13
```

The server will accept other options to control how Prep builds the table. The "Method" and "Interval" qualifiers control the interpolation system. Prep will use the "Write All Values" setting if you don't give a Method qualifier. Valid values for the method are:

Interpolate	Fixed interval interpolation
Average	Compute the average over the interval
Total	Compute the total over the interval, for incremental data
Moving Average	Compute a moving average over the interval

An example is:

```
http://host/data.hts?Service=Hilltop&Request=DataTable&
Site=Alphabet River at Crossword&Measurement=Flow&
Method=Average&Interval=1 day&
From=1/1/13&To=1/7/13
```

Prep can write the location of the site using its easting and northing. Use the "Location=Yes" key-value pair to see the location in the table.

Ensemble Statistics

Hilltop Hydro can draw envelope plots and the P3 command will list the numbers that are drawn on the plot. Hilltop Server can run the same calculations and return the results in an XML document. The keywords are similar to those used on the GetData command but the time range defaults to all of the data instead of the most recent value. The From, To and TimeInterval keywords are still available and will set the time range as given. The following example will compute a monthly probability density function for all the data:

```
http://host/data.hts?Service=Hilltop&Request=EnsembleStats&Site=Alphabet
    River at Crossword&Measurement=Flow&Statistic=MonthlyPDF
```

The Statistic keyword determines the analysis method on the data and valid settings are as follows. Consult the on-line help for P3 to get a complete explanation of the possible choices. You need to have a clear head to understand each one, but the PDF is the simplest one to go for.

HourlyExtrema	MeanHourlyExtrema	ExtremeHourlyMean	HourlyPDF
DailyExtrema	MeanDailyExtrema	ExtremeDailyMean	DailyPDF
MonthlyExtrema	MeanMonthlyExtrema	ExtremeMonthlyMean	MonthlyPDF

All statistic options give the min, mean and max, and the PDF options give the standard deviation, median, and values at two percentiles that you specify. You get the 20th and 80th percentile if you don’t give a keyword. An example to fetch the 5th and 95th percentiles is:

```
http://host/data.hts?Service=Hilltop&Request=EnsembleStats&Site=Alphabet
    River at Crossword&Measurement=Flow&Statistic=MonthlyPDF&
    LowerPercentile=5&UpperPercentile=95
```

OGC Services

The Hilltop Server accepts OGC compliant requests. The server provides basic documentation for the OGC commands, and this may be helpful to people who wish to fetch data from your server. Get them to point their web browser at the Hilltop Server with a nominal Hilltop file name. This will return an HTML page. For example:

```
http://host/help.hts
```

Web Feature Service

Information about the available data is returned using the Web Feature Service, WFS. The Hilltop Server implements three requests, being GetCapabilities, DescribeFeatureType, and GetFeature. The capabilities request lists the available feature types, and also describes the very limited filters the server will apply. The DescribeFeatureType gives details for the feature types. You can see the response from these requests in a web browser, as follows. Neither of the commands takes any other parameters.

```
http://host/data.hts?Service=WFS&Request=GetCapabilities
http://host/data.hts?Service=WFS&Request=DescribeFeatureType
```

The SiteList feature type is a list of sites, hence the name. Each entry in the list is the name of the site and its location. The MeasurementList feature type expands on the site list, but gives the measurements at the site, and the time range of the measurement.

The GetFeature request sends the data for the feature identified in the TypeName key. The request will also accept a bounding box.

```
http://host/data.hts?Service=WFS&Request=GetFeature&TypeName=SiteList
http://host/data.hts?Service=WFS&Request=GetFeature&TypeName=MeasurementList
http://host/data.hts?Service=WFS&Request=GetFeature&TypeName=SiteList&
    BBox=-46.48797124,167.65999182,-44.73293297,168.83236546
```

The BBox key accepts two latitude and longitude pairs. The order of the pairs is not important, but the latitude precedes the longitude in each pair. The server needs to know which projection is being used in the Sites table, so make sure you have a Projection entry in the HilltopSystem.dsn file. Examples are:

```
[Hilltop]
Projection = NZMG              ; The old system, NZ Map Grid
Projection = NZTM2000          ; The replacement system
```

You can also specify the bounding box in NZMG or NZTM2000. The proper OGC method is to have a Universal Resource Name at the end of the BBox value. The URN should be crafted in accordance with OGC 07-092, or else, and examples are:

```
urn:ogc:def:crs:EPSG::4326    WGS84 lat and long
urn:ogc:def:crs:EPSG::2193    NZTM 2000
urn:ogc:def:crs:EPSG::27200   NZMG
urn:ogc:def:crs:EPSG::4167    NZGD 2000
```

Thus a complete BBox entry using easting and northings in NZMG is:

```
BBox= 2100000,5400000,2180000,5600000,urn:ogc:def:crs:EPSG::27200
```

The server will convert the co-ordinates of the bounding box into the co-ordinates of the Sites table, and then do the search.

Lawa Table F

The server can prepare the data required for Table F in the Lawa data request. The type name was originally a nice simple name, but now is “MonitoringSiteReferenceData”, spelt as shown. Thus a URL to fetch the Table F is:

```
http://host/data.hts?Service=WFS&Request=GetFeature&TypeName=MonitoringSiteReferenceDa
ta
```

Sensor Observation Service

The Hilltop Server uses Sensor Observation Service, SOS, to transmit the data. The SOS implementation is restricted and the server only accepts the GetCapabilities and GetObservation requests. The SOS service doesn't provide any meta-data, and only sends the data in WaterML 2.

The GetObservation request accepts a site name, measurement and time range. The names are of the keys are:

Site Name	FeatureOfInterest
Measurement	ObservedProperty
Time Range	TemporalFilter

Setting the Time Range

The TemporalFilter key is a Time Interval as defined in ISO8601 and full details are available at http://en.wikipedia.org/wiki/ISO_8601 . You can provide both a start time and a finish time, or just a start time and the server will read until the end of the data. The server will give the most recent value if you don't provide a temporal filter at all.

Examples are:

```
TemporalFilter=om:phenomenonTime,2010-01-01T12:00:00/2011-07-01T14:00:00
TemporalFilter=om:phenomenonTime,2010-01-01T00:00
```

You can also give a duration, and this can be done in place of either the start or the finish time. A duration in ISO8601 has the letter 'P' then one or more pairs of numbers and letters. The letters refer to the units, and are obvious choices. The letter 'M' is used for both months and minutes so the duration must have the letter 'T' in it before the time of day units are used. Examples are:

P1Y	One year
P10Y	Ten years
P3M	Three months
P3D	Three days
P1Y3M	One year and three months
PT3H	Three hours
PT10M	Ten minutes
PT30S	Thirty seconds
P3MT3H30M	Three months, 3 hours and 30 minutes.

Examples are:

```
TemporalFilter=om:phenomenonTime,2010-01-01T12:00:00/P1Y
TemporalFilter=om:phenomenonTime,P1Y/2011-01-01T00:00
TemporalFilter=om:phenomenonTime,P7D
```

The first example gives a year of data from the given start time, and the second example gives the year before the given finish time. The third example has a duration by itself and this taken as interval back from the end of the available data. In the example, the command will return the seven days up to the most recent date and time.

An example of a GetObservation request is:

```
http://host/data.hts?Service=SOS&Request=GetObservation&
    FeatureOfInterest=Alphabet River at Crossword&
    ObservedProperty=Stage&
    TemporalFilter=om:phenomenonTime, 2012-01-01T00:00/2012-11-01T00:00
```

Clients

You can write your own client to talk to the server or use Hilltop clients. The Hilltop clients still think they are reading data from a file, but of course it’s now a File DSN. The File DSN has a style of HTServer and contains the URL to get to the server. You need to provide the path, with an optional port number, and a filename with the filename extension that is mapped to the server. An example is:

```
[Hilltop]
Style=HTServer
URL=http://host/Data.hts
```

Don’t put any key-value pairs on the URL, the Hilltop clients will do this for you.

Hilltop clients will pass the site and measurement names through the Hilltop Transfers table. This means you can arrange to see the data using names of your choice. Create an Agency that matches that sent by the server, and then define substitute site names and measurements as usual. You can see the agency for the server in Hilltop Manager. Open the File DSN in Hilltop Manager and the agency will be shown in the results pane on the right hand side of the window.

Web Sites

Hilltop Server is designed to work with a web site and will help make your web site configurable and dynamic. You will need to write software to drive the site and your web site designer will choose to have this software run on a Web Server or on the customer's computer. This is called "Server Side" or "Client Side" programming.

Server Side software is mostly likely written in either C# or PHP. Client side software is written in JavaScript. All languages work as well or as badly as each other, and designers love to debate the relative merits.

Client side software in JavaScript is slightly more complicated because JavaScript loves a data transfer language called JSON. JSON and Javascript play together well and it is very easy to take a JSON object or array and put it into the equivalent structure in Javascript. Real JavaScript programmers prefer JSON to XML, and that is saying it politely.

The OGC standards are mostly written for XML, and while this is changing slowly, the responses from Hilltop Server will be in XML for some time yet. The easiest way to get a response from Hilltop Server into a Javascript object or array is to use the jQuery library. The following code snippet shows how to ask a server for a site list and then load the list into a JavaScript array.

```
var SiteNames = [];
$.ajax({
        type: "GET",
        url: "http:/host/data.hts?Service=Hilltop&Request=SiteList",
        dataType: "xml",
        success: function(xml) {
                $(xml).find("Site").each(function() {
                        SiteNames.push($(this).attr('Name'));
                });
        });
});
```

The code calls the jQuery Ajax function and passes it an object that has information about the request. In Javascript, items in an object are enclosed in braces. The "url" item has the address, and the "dataType" item tells jQuery to parse the response as XML. The "success" item has a function that will run once the response is received and put through the XML parser. The function accepts an object that contains the xml parser object, and the function can then fetch elements from the XML. The "$(xml).find()" function builds a collection of the Site elements in the XML, and then function defined in the ".each()" part of the statement is called for each Site element. The function fetches the site name and pushes it into the SiteNames array. This code is not pretty, but it is concise and will run quickly enough.