OPC Explorer

SAE – Automation, s.r.o. Nová Dubnica
Solid **A**nd **E**ffective partner at development of **your** products and industry automation

Internet browser based OPC client

Web technologies offer easy access to OPC servers

Web based technologies and their simplicity find a way to more and more applications. Nowadays, browsers are capable to perform various tasks and web based applications are reaching the quality and behavior of standard desktop applications as known today. It is only a question of time when browsers emerge with desktop and take the place of an operating system and will able to perform all desktop tasks.[1]

This model assumes there is a web service available that provides the access to a OPC server. The web service ensures the data exchange between the client (web browser) and OPC server. Retrieved data from the OPC server flow through the web service and are passed to the client browser that displays them in a HTML page. The communication between the web service and OPC server is based on the OPC DA protocol specification that ensures transparent data exchange between two OPC clients. The question was how to pass the data to a web based client in a suitable machine-readable form that complies with HTTP protocol. **XML DA specification** solved this problem and offered a solution how to exchange data between a web based client and a OPC server.

Communication model with the use of a web based client

When using a web based client, the traditional server-client architecture can't be used anymore. A third element must be involved – a XML DA wrapper, web service (wrapper) that ensures the data exchange between OPC server and the client (see Figure 1).

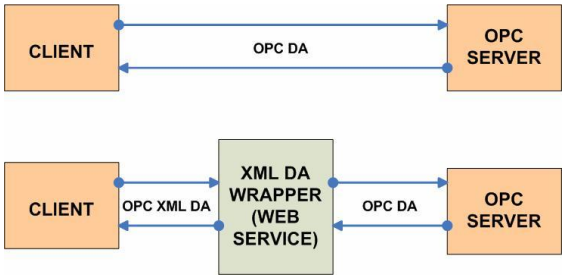

Figure 1: Communication model and data exchange

The wrapper serves as a proxy during the communication between the server and a web based client, it transforms data into readable form by the client and sends them over a HTTP protocol. For this purpose the wrapper should comply with the XML DA specification as published by OPC Foundation[2] to fulfill best practices and easy interoperability in data exchange.

Web 2.0 improves the Web based data exchange

With the upcoming, so called, Web 2.0 it is possible to create powerful Internet browser based application that offers better scalability and reduces the traffic while exchanging data between a client and a web server. This is possible via AJAX technology (Asynchronous JavaScript And XML)[3]. During a normal HTTP request-response communication entire web page and HTML is transferred from the server to the client. With the use of AJAX and XML it is possible to reduce the traffic and transfer only parts that need to be updated. In our case it means we only exchange the data that originate from an OPC server and pass through a XML DA web service (see Fig. 2)

[1] http://www.pyrodesktop.org/Main_Page

[2] http://www.opcfoundation.org/DownloadFile.aspx?CM=3&RI=231&CN=KEY&CI=278&CU=1

[3] http://en.wikipedia.org/wiki/Ajax_%28programming%29

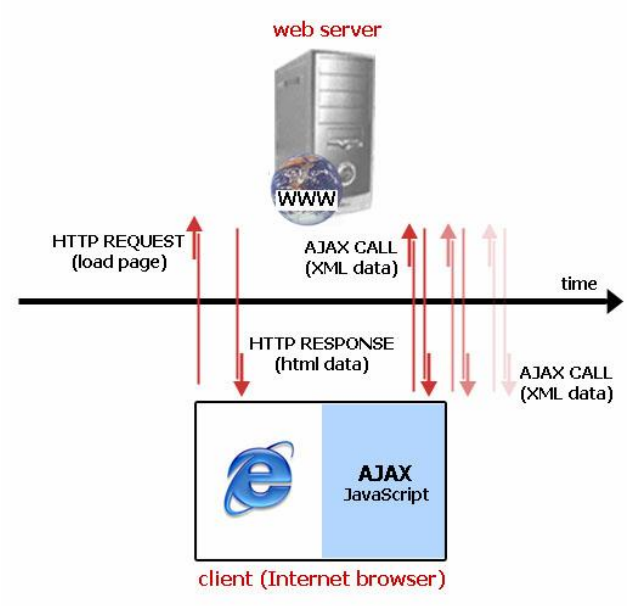

Figure 2: Scheme of HTTP traffic using AJAX calls

Full HTML is loaded only at the very first time a user visits the web page. The HTML content is served in its full range. However, after the web page is displayed and structured, the AJAX engine inside a web browser takes place and makes calls back to the web server to fetch updated data and display them in the browser. This way we refresh only OPC data that changed, keeping old data intact and thus reducing the overall traffic that would take place in a standard synchronous request-response HTTP model.

OPC EXPLORER in action

OPC Explorer application is written in ASP.NET language. It consists of two parts:

- a interface that displays the data (index.aspx)
- web service that translates the data to and from the client (services.aspx)

The interface is responsible for data visualization. It contains the AJAX (JavaScript) engine that makes calls to the server and updates data. It communicates directly and only with the OPC Explorer web service.

The web service then reformats the data and sends them to XML DA wrapper and respectively. When the client (interface) requests the update of data, it sends a HTTP request with simple XML data to the web service (this simple XML document specifies what data should be updated). The web service then reformats these data into XML DA specification format and passes them to the XML DA wrapper. The XML DA wrapper then fetches the data from the OPC server and sends them back to the web service. Again the web service unwraps these data and reformats them to a simpler XML document that is readable by the client interface. The interface parses these simple XML data and outputs them as HTML to browser. The reason why OPC Explorer web service is used is that browsers have limited in-built XML support so we moved this critical section to the web server. Of course, the browser functions grow rapidly and the need of OPC Explorer web service will cease in the future.

Of course, this model has its main advantage when the OPC Explorer web service and XML DA wrapper are served by the same web server.

OPC Explorer features

- easy configuration through a single XML file
- fetch data from OPC server via READ or SUBSCRIBE method (to read items from OPC server)
- write data to OPC server (to write new OPC item values)
- data visualization and graphs[4]

OPC Explorer is not meant to be an all-in-one application. It is designed to watch OPC items and visualize the data. Development and adding of new features, binding to SAE – Automation, s.r.o. products (SAEAUT SNMP OPC Server and OpcDbGateway), better data handling, visualization and allowing configuration of OPC servers is in progress.

Using OPC Explorer

Installation

Software prerequisites to use with OPC Explorer on the server:

- Microsoft IIS 5.0 or higher
- Microsoft .NET Framework 1.0 or higher

[4] May not be included in product by default. Particular visualization implementation depends on individual customer's needs and wishes and is the subject of product modification.

- XML DA wrapper (for example ICONICS XML DA Wrapper or Advosol XML DA Gateway)

Software prerequisites to use with OPC Explorer on the client:

- Internet browser (for example Internet Explorer or Mozilla Firefox)

No other software is required to run OPC Explorer. From the client point of view there is no software installation needed because OPC Explorer runs in the browser on a demand – a user has just to visit a the OPC Explorer website on the web server.

Configuration

The OPC Explorer configuration settings can be found in single XML file (config.xml) enclosed in XML tags:

- **ServiceURL –** sets the URL of XML DA wrapper web service to which OPC Explorer should connect
- **UpdateRate** – defines how often in seconds data should be retrieved from XML DA wrapper (default value: 5)
- **UseSubscription** – defines the method to update data from XML DA wrapper, set to **true** for SUBSCRIBE method or **false** for READ method (default value: true)
- **ID** – name of the OPC item to read, to add multiple items enclose the OPC item name between <ID> and </ID> tags (e.g. <ID>Device.System.DescInfo</ID>)

Operations with OPC Explorer

The connection to OPC server is easy. After the service URL is configured the user just presses the "Connect" button. OPC Explorer then contacts the OPC server and reads data from it. If Subscription method is allowed and no data are available yet (first read is being performed) then OPC Explorer automatically creates a subscription and uses this subscription reference for further connections. Read items are displayed in a nice transparent form (see Fig. 3).

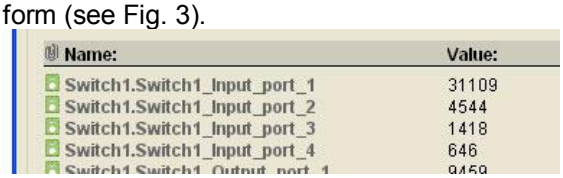

Figure 3: Viewing item values

The data are updated regularly according to configured UpdateRate in seconds. OPC Explorer handles subscriptions method automatically – a subscription is created, refreshed and cancelled according to user's actions.

The user has the option to switch between "values" or timestamp and OPC connection quality (see Fig. 4) details by clicking on the table header. The connection quality is also indicated by the green (good connection) or red (otherwise) icon left to the item name. This way the user can easily see all important details about an OPC item. The color of item names (black or gray) also distinguishes the age of particular item. If an item has been refreshed in last performed update then the item name color will become black, the item name color stays gray otherwise.

Figure 4: Timestamp and quality

It is possible to update values of listed OPC items through OPC Explorer by clicking on the OPC item name and assigning a new value in the dialog window that appears afterwards. After pressing the "Write" button the new value is sent to OPC server and a result message appears if the operation finished successfully or with an error.

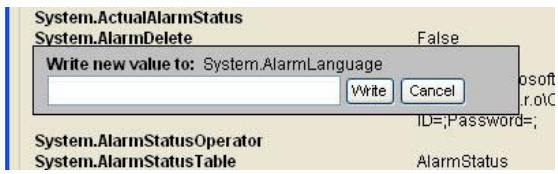

Figure 5: Writing new OPC values

Data visualization

The data visualization engine is a subject of OPC Explorer modification. There is no default way how to visualize received data. SAE - Automation, s.r.o. modifies OPC Explorer according to individual demands and needs to fit customer's wishes at best.

Example data visualization is shown at figure 6 that is used in the Networking Monitoring demo application. In this case the visualized data represent the bandwidth usage of particular computers and routers. This way a operator

can easily detect lack of bandwidth or unusual network behavior. Of course, the visualization styles can be modified in various ways to fit one's needs and wishes.

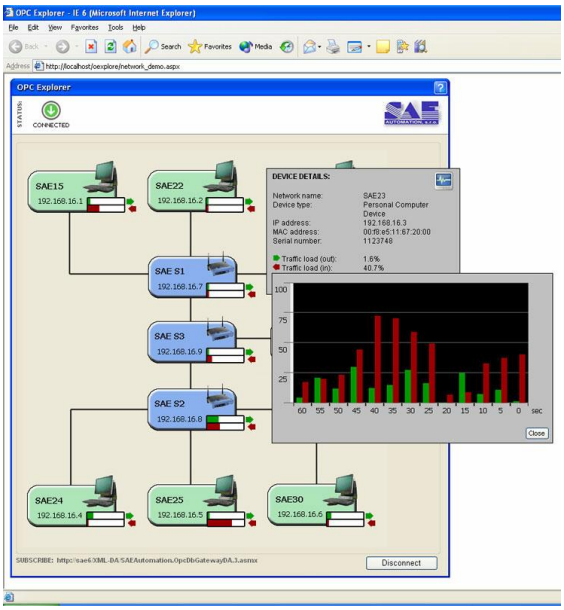

Figure 6: OPC Explorer in Internet Explorer with the look & feel of Windows XP – data visualization

Example case study

In our example we used model as follows (see Fig. 7)

- several SNMP devices send various data about their status and operation via SNMP protocol
- our own product SAEAUT SNMP OPC Server receives the data sent from SNMP devices and processes them into OPC items
- our another product OpcDbGateway that collects data from several OPC servers (in our case, it collects data just from 1 server but it is possible to attach OpcDbGateway to more OPC servers - and each of the OPC servers can be connected to several SNMP devices, etc. so a very complex architecture can be created) - this way we created just one single interface, that we will access later, and thus simplified the process of data exchange between XML DA web service and OPC servers)
- XML DA wrapper/web service (we used ICONICS XML DA wrapper) is connected to OpcDbGateway and provides a web based access to OPC servers
- because the XML DA specification is too complex, we created another web service (OPC Explorer web service) that translates the data from the full XML DA specification form to a more simple XML document, thus reducing the amount of traffic required when exchanging data between the web service and client
- at last, the client (Internet browser) receives the data and displays them as a HTML document, the browser doesn't connect directly to XML DA wrapper but to another web service that transforms all response data to a simpler form and reduces the HTTP traffic)

What is exactly happening?

1. A user starts his favorite web browser and visits OPC Explorer website.
2. The browser sends a request to the OPC Explorer web service.
3. OPC Explorer web service reads the data, wraps them into a XML formed document (as defined in the XML DA specification) and passes them to the XML DA wrapper.
4. XML DA wrapper sends a request to OpcDbGateway and performs all steps that are necessary to take place (eg. read data, write data, get server status, etc.)
5. Updated data from OpcDbGateway are passed back to the XML DA wrapper. XML DA wrapper sends them back to OPC Explorer web service as a XML DA formed document.
6. OPC Explorer web service unwraps received data and formats them into a simple XML document that is easily readable by the web browser. The data are then sent to the browser.
7. The browser displays the data as HTML.

Considerations

To achieve less traffic in the communication between the client and the OPC server we use another go-between – OPC Explorer web service. If both web services (OPC Explorer and XML DA wrapper) are hosted on the same server, less traffic occurs in the entire communication between the client and the OPC server. The other reason why the client

(browser) does not parse the XML DA response from the wrapper directly is that browsers have limited capability of parsing XML documents. The XML DA specification is quite complex and that's why we use OPC Explorer web service to reformat the data to a simpler format. Of course, as capabilities of browsers grow constantly, the need of OPC Explorer web service as a go-between may cease in the future and this part will be implemented directly in the interface.

If the OPC server, XML DA wrapper and OPC Explorer are hosted on the same server then we achieve a great traffic reduction and a slight speedup of entire application. In this case, the data flow physically only between two computers (the client that runs web browser and the server that hosts the OPC server and both web services). The OPC server, XML DA wrapper and OPC Explorer web service communication does not occur over network and thus speeds up the entire communication.

While deploying an Internet browser based OPC client, cross-browser compatibility issues must be taken into consideration. It is necessary to test the application against more browsers and ensure its proper functions.

Web browser based OPC client requires only a web browser – no other software is required to install. Software upgrades are performed just on the server without the need of any client installation or configuration.

1. Conclusion

We achieved to develop an Internet browser based OPC client that can read and write data to and from an OPC server and visualize the data. This is reached by using of JavaScript and AJAX framework (OPC Explorer utilizes jQuery Javascript framework with included AJAX library)[5]. Asynchronous approach to the traditional HTTP request-response communication model allows us to reduce the network traffic that is necessary to take place. Another advantage is that the application interface and data visualization is common for all clients and cross-browser compatible. Because a web application is served by one server this offers also easy application upgrades and changes to the application configuration without the need of separate configuration on client machines.

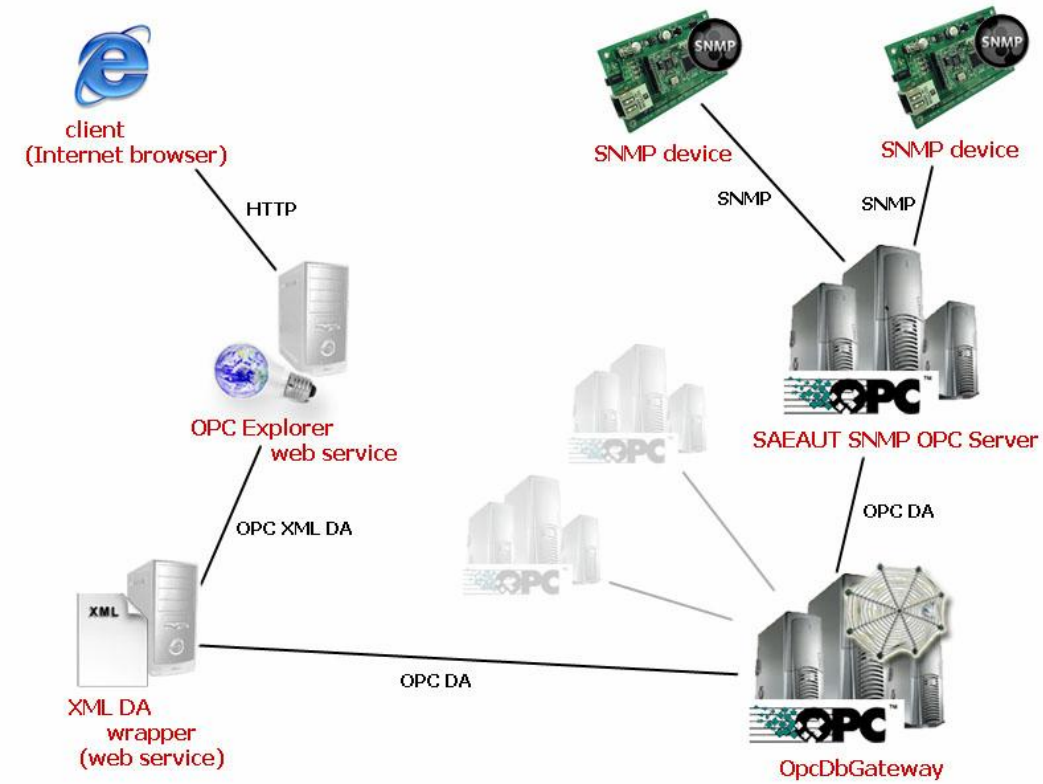

Figure 7: Example case study

[5] http://www.jquery.com