Updating a UNIX Application Suite for the Windows NT World

Thomas W. Hutchinson

Ronald R. Derynck

A project team learned some useful lessons in porting a real-time software platform for industrial applications to an environment that typically runs desktop applications such as word processors, database programs, and spreadsheet applications.

One of the remarkable trends in software in the 1990s has been the influence of Microsoft® technologies on many industries. The area of industrial automation and supervisory control is no exception. HP has updated a successful UNIX®-based product to respond to this trend while maintaining many of the product's strengths. The product, HP RTAP (real-time application platform), is an example of a real-world industrial application of component software technology.

Background

HP RTAP is a complete software framework for building industrial automation systems. It is used in a wide variety of industries and is sold mainly through channel partners, such as system integrators, who focus on a particular industry or geographical area. HP RTAP has been a successful product and has been available on the UNIX operating system since 1990.

Figure 1 shows the architecture of the UNIX-based version of HP RTAP. The core of HP RTAP is a high-performance real-time process database based on object-oriented concepts. It contains a spreadsheet-like calculation engine. It also has a master timekeeper, a scan system for remote devices, and facilities for managing events, detecting alarms, recording historical data, and determining the health of the system.

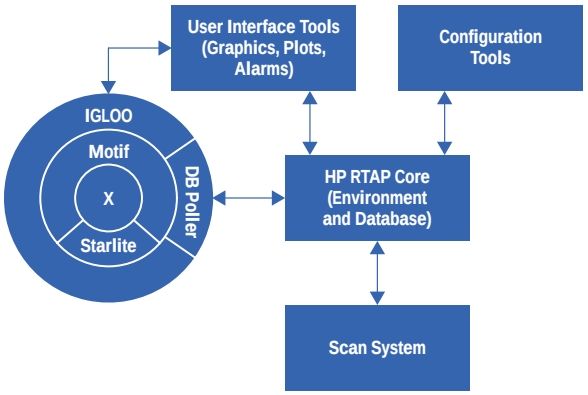
HP RTAP also contains a complete graphical user interface (GUI) including alarm and trend displays, reports, schematics, control panels, and a suite of

Thomas W. Hutchinson
An R&D project manager at the HP Calgary Product Development Center, Thomas Hutchinson is responsible for software development of HP RTAP products. He has been with HP since 1987. He received a BSc degree in computer science in 1979 from the University of Calgary. Tom was born in Taber, Alberta, Canada. He is married and has five children. In his leisure time he enjoys hiking, camping, cross-country skiing, and researching family history.

Ronald R. Derynck
Ronald Derynck is an R&D section manager at the HP Calgary Product Development Center. He was the R&D manager for the HP Enterprise Link and HP RTAP development programs and is now the business team leader for both of these products. He holds BSc (1970) and MSc (1972) degrees in electrical engineering from the University of Calgary. He came to HP in 1986. Ron was born in Calgary, Alberta, Canada, and his interests outside of work include biking and cross-country skiing in the Canadian Rockies.

Figure 1

The current UNIX-based HP RTAP architecture.



configuration tools for core and graphical components. The GUI is built on an HP graphics library that is provided for customers to build their own graphical applications. The GUI library, in turn, is built on X-Windows and Motif. Early versions of HP RTAP were on the X10 windows system. Later, ports were done to X11 and Motif. Customers were protected from changes because they were hidden inside the HP graphics library.

The HP RTAP product was first introduced on the HP-UX operating system and soon became known for its open systems focus. It is now supported on several of the most popular UNIX platforms.

This article describes some of the modifications we made to the HP RTAP product to port it to the Windows® NT environment. The goals we had for this project were to:

- Offer the Windows NT market a supervisory control and data acquisition system with the power and flexibility of HP RTAP
- Enhance the usability of the UNIX product by allowing full integration with the Microsoft desktop on the client side
- Support HP corporate goals for growth and integration of both UNIX and Windows systems, taking advantage of the strengths of each.

Moving to Windows NT

Microsoft Windows NT has recently been gaining acceptance as a platform for industrial automation systems. However, as is often the case, there is a difference of opinion among users. Some users with mission-critical applications are not ready to move to Windows NT. The UNIX system, on the other hand, has been on the market for over 10 years and is a mature, stable, and robust operating system. Other users are ready to embrace the Microsoft world fully because this environment better satisfies their need to incorporate real-time information into the decision making process. Still others want to keep the UNIX operating system for their servers and provide the benefits of the Microsoft world on their desktops.

We considered all this information in our plans for a product on Windows NT. Our first major decision was to split the product structure along client/server lines. There was general agreement that since Microsoft dominates the desktop world, HP RTAP needed a Windows interface. On the other hand, opinion was still divided on the server side, so we chose to support both UNIX and Windows NT operating systems for servers and allow our Windows clients to connect to both.

HP RTAP is well known and respected for its core data server, so it was ported to Windows NT with existing functionality but with a different graphical user interface. Although third-party products are available to allow a port with the Motif look and feel, we decided to rebuild

Glossary

ActiveX. A COM-based framework for software component integration. An ActiveX Control is a prebuilt, reusable software component that performs a particular function (often visual and interactive) within the context of container applications such as VisualBasic or a Web browser.

CORBA (Common Object Request Broker Architecture). An object request broker that provides the services which enable objects to make and receive requests and responses in an object-oriented distributed environment.

COM (Component Object Model). A binary programming interface standard that allows components written separately (even in different languages) to communicate correctly.

DCOM (Distributed Component Object Model). The COM distributed wire protocol.

OLE (Object Linking and Embedding). Windows' compound document protocol that allows one document to be embedded within or linked to another document.[1]

parts of the HP RTAP user interface and purchase other parts based on the new Microsoft model for component software.

With this philosophy, we created a graphical user interface based on Microsoft COM (component object model). It provides powerful drawing and animation capabilities, Visual Basic for applications, and a set of industrial ActiveX controls that users can supplement by adding their own components. For example, a user can select an ActiveX control that looks and acts like a gauge to keep track of filling a vessel. We also added a new configuration tool that leverages from available Microsoft usability features and emphasizes the class-based nature of the HP RTAP database. This allows complete integration with other tools on the Microsoft desktop. All client-side tools use a new ActiveX component to communicate with the HP RTAP server on either Windows NT or UNIX operating systems.

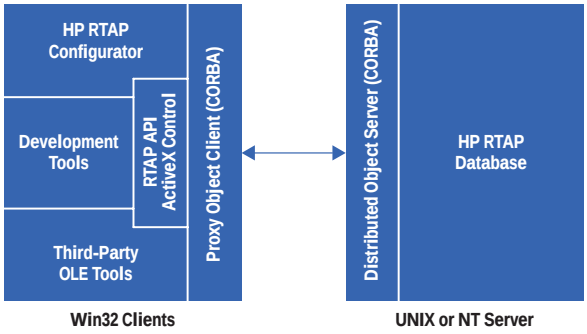
In addition to ActiveX, another important Microsoft technology is the ODBC (Open Database Connectivity) standard for relational databases. Users want to create reports using their regular spreadsheet, database, or word processor tools. They do not want to learn a special report generator. The new HP RTAP ODBC interface takes advantage of the class-based nature of the HP RTAP database, allowing useful queries of similar points as if they were relational tables. For example, HP RTAP will now accept an SQL SELECT statement for a query like "show me the site locations for all gas wells having a flowing temperature greater than 20 degrees Celsius."

We emphasize that the PC client-side environment is truly a Microsoft world so that all the RTAP PC clients integrate perfectly with Visual Basic 5, Visual C++ 5.0, Delphi, Access, Excel, and all the other tools and applications users expect. Access to the HP RTAP database and the rest of the server environment is through ActiveX components and ODBC. Users get all the benefits of integration on the desktop and access to the servers on either NT or UNIX systems. This allows existing customers with UNIX-based HP RTAP systems to add the PC clients whenever they are ready.

An important building block that enables this client/server split is the HP RTAP API ActiveX control (see **Figure 2**). This ActiveX control is the mechanism by which all PC clients connect and communicate with the server. All HP RTAP database components are represented as objects

Figure 2

New HP RTAP clients based on Microsoft OLE.



with methods and properties like any other ActiveX object. This enables the full power of the HP RTAP server-side API, but with a much simpler object-oriented interface. For example, **Figure 3** shows a simple Visual Basic code fragment for reading a database value. The API supports more than simple reads and writes. It allows for more complex operations such as caching and large grouped data transfers. We chose to create a full object model, allowing advanced functions such as configuration of the database from a PC client through the ActiveX interface.

The interface seen by the user of the HP RTAP ActiveX control is the familiar OLE-style interface seen in Visual Basic and other desktop tools. Thus, a user, in say Visual Basic, sees an OLE style API. On the other side of the ActiveX control (hidden from the user) is the infrastructure for communicating with servers. We used CORBA as the underlying platform because of its portability and maturity.

Climbing the Learning Curve

Connecting PC clients was the largest technical challenge that we faced in this migration project. Our development team had years of UNIX experience, but ActiveX and CORBA were new to us. We hired some local contractors to alleviate this problem, but our collective learning curve was still quite steep. The job turned out to be much bigger and more difficult than we or the contractors expected.

One major challenge was to create an object model that would represent all the aspects of HP RTAP and be exposed to users (for example, database points and attributes). The interface had to be consistent and simple to understand but powerful enough to support configuration

Figure 3

Simple Visual Basic code fragment for reading a database value.

```
'General declarations for the RTAP environment & db objects
Dim g_env as HPRTAPLib.Environment
Dim g_db As HPRTAPLib.Database
Dim g_pt As HPRTAPLib.Point
Dim g_attr As HPRTAPLib.Attribute

'Open a connection to the environment and the database when the
'form is loaded. The environment name and host computer are
'specified in the properties of the RTAP Custom Control.

Private Sub Form_Load()

      'Connect to the RTAP environment
      Set g_env = HpRtap1.Environment
      g_env.Connect

      'Connect to the database and read the current value
      If g_env.Connected Then
            Set g_db = g_env.Database
            Set g_pt = g_db.PointByAlias("PT-101")
            Set g_attr = g_pt.Attribute("process value")
            Label1.Caption = g_attr.Value
      End If
End Sub
```

and fast access to values. The resulting object model has about 40 classes and over 300 methods and properties.

After the object model was defined, we had to implement it in the ActiveX control and CORBA layers. Aside from learning how ActiveX and CORBA work, developing this code was not particularly difficult. We were familiar with the HP RTAP API, and programs that receive CORBA requests on the server side were fairly straightforward to write and implement.

Another challenging area was the memory allocation model. Both COM and CORBA base their memory schemes on the concept of reference counts. In principle, it is simple: for every created object, a counter is maintained on how many times the object is used, and when that count drops to zero, it is safe to delete the object. However, in practice it is not so simple, and an error can have serious consequences. Freeing an object too soon can cause invalid memory references later, leading to a disastrous failure of the program. Failing to free the object can create a memory leak, which leads to a slower death but is still fatal in a program that must run for weeks or months. In particular, CORBA implementations do not automatically forward or synchronize reference counts between the client and the server, which became quite difficult for us to manage.

Our first implementation uses CORBA because it is a mature technology and is available on Windows NT and all the UNIX platforms that we support.

Scanning the Results

Porting the HP RTAP server to Windows NT maintains all its UNIX functionality and existing API. It should be noted that this server, even on a PC, still retains some of its UNIX behavior. We believe this is a reasonable trade-off because it provides the following advantages:

- It maintains compatibility with UNIX servers, allowing the same clients to work with both.
- It gives customers a migration path for their current and future server-side applications.
- It provides scalability that PC-only software cannot match.

These modifications to HP RTAP met our goals for the project.

Conclusion

This migration from UNIX to Windows NT has taught us a lot about new technologies that will be useful in future development efforts. In particular, we learned two surprising lessons about this type of project.

First, it is not any easier to develop complex software on Windows NT than it is on a UNIX operating system. There is a tremendous framework available for making applications easier for users to use, and the infrastructure is definitely improving for developers as well—at least for certain kinds of applications. For the kinds of industrial automation software that we write, our development schedules cannot be made significantly shorter just because we are on Windows NT.

Secondly, we have learned useful lessons about using third-party software to develop applications. We have gone from an environment in which we are in control of everything to one in which we rely on many pieces of software from at least six vendors (not including Microsoft). This has many implications for release scheduling, documentation, and reliability of the end product. It has caused problems and will undoubtedly continue to do so. However, in the final analysis, we are convinced that there are many gains to be made from Microsoft's component software model and from generally being able to use software created by others.

Reference

1. A. Freedman, *The Computer Glossary*, AMACOM, 1995, p. 276.

Online Information

Additional information about HP RTAP is available at:

http://www.hp.com/go/ais