August 2001

# IBM Wireless Solution for Field Force Automation

Chris Pavlovski
Tony Kambourakis

## Highlights

- ***FFA Solutions provide considerable productivity gains which translate into real cost savings.***
- **Potential for improved revenue and profit by pro-actively managing faults to minimise service outages.**
- ***Minimise impacts to customers by providing real-time status information.***
- ***Potential reduction in IT costs through the use of less expensive devices.***
- ***Scalable, manageable and secure solutions.***

### *Wireless Computing: A Mature Market[1]*

What is wireless computing? Wireless Computing is convenient access through a new class of appliances, to relevant information with the ability to easily take action on it, when and where you need it.

The number of wireless (or mobile) devices is expected to multiply rapidly over the next few years. Market research has predicted that by 2003 the number of pervasive computing devices will exceed the 6 billion estimated number of people worldwide[2]. Specifically, there will be more than 300 million personal digital assistants (PDAs), 2 billion consumer electronic devices such as wireless phones, pagers and set top boxes, and 5 billion additional everyday devices like vending machines, refrigerators and washing machines embedded with chips connected to the Internet. Companies must be prepared not only to extend the corporate infrastructure to meet the demand, but to modify their existing applications so that devices become completely integrated into existing mission-critical systems.

IBM's strategic intent is to provide end-to-end wireless e-business solutions with industry partners and to promote open standards.

#### *New uses for existing data possible today*

The rapid proliferation of mobile devices has been obvious for some time now. However, only recently has the convergence of new Internet technologies, such as Web Application Servers, Extensible Markup Language (XML), and Lightweight Directory Access Protocol (LDAP), together with mobile technologies such as the Wireless Application Protocol (WAP), made it practical to extend workforce management information out to mobile user populations. With a standards-based approach, organisations can feel comfortable that their investment in information systems is protected.

These technological solutions and advances can now be applied to solve well known business problems in a pragmatic way, that enables the organisation to deploy enterprise solutions to its staffing pool, business partners, and customers.

#### *Portable Digital Assistants: Mobilising the Workforce*

This proliferation of mobile and wireless technology has led to a change in the way corporate and consumer based applications are developed and accessed. The enterprise network can now be extended to a fully mobile workforce through a combination of wireless technologies, networks, and portable devices.

By extending the existing information systems out to mobile workers, organisations can leverage the existing IT infrastructure and in turn, they increase the value of that infrastructure through more timely information from the field.

The deployment of PDA (Personal Data Assistance) devices within an organisation, has become a feasible alternative to the desktop or laptop computer. Users vary from managerial staff accessing e-mail, calendar, and scheduling functions, to field workers accessing legacy applications such as a Job Dispatch and Reporting system. These devices also provide several attractive advantages over traditional computing platforms. These advantages include a significantly lower cost in terms of the device itself, and the software platform required to support the functions users demand, commercial off-the-shelf product that will be supported by a range of software tools, and with increasing processing power available as the device matures. In particular, when one considers other forms of mobile devices such as proprietary hand-held devices, the costs associated with maintaining and upgrading these proprietary devices increases with time, whilst the processing power remains stagnant. Furthermore, such proprietary devices may inevitably become unsupported. All these factors contribute to the attractiveness of deploying solutions that employ to commercial PDA devices.

### *Business Case Benefits of Field Automation*

Most companies presently bill its corporate customers for services performed and materials used. In an increasing competitive market, commercial customers are progressively requesting justification for the expenses associated with the services performed. This will be in the form of substantiating at a detailed level, all activities performed with their associated cost. In general, manual or semi automated systems for capturing this detail are inappropriate and may hamper the business's ability to capture all revenues due to them. This is primarily due to the manual capture methods currently in practice.

The Field Force Automation solution will provide field engineers with tools for capturing such data at the time when the work is performed, providing greater accuracy. Furthermore, the tools can be customised so that the specific billing justification requirements of the business may be met.

### *Improved Revenue Collection Methods*

As suggested above, the current revenue collections process may generally involve a manual process. Specifically, field engineers will typically write down activities performed onsite. In many cases, the accuracy of capturing works performed is inadequate, furthermore, the notes written down need to be entered into a computing system. This is typically completed at the end of the week, and is complicated further when the shorthand used by field engineers is difficult to read – contributing to further inaccuracies.

The Field Force Automation solution is able to provide a comprehensive list of items or activities performed, so that all that is required is for the field engineer to check the appropriate work activity. Furthermore, the next time the field engineer synchronises the PDA, the information is transferred to the legacy (or core) system, enabling billing to be initiated in a more timely fashion.

### *Capturing Planned and Unplanned Work*

In addition to performing scheduled, or planned activities the field force engineer is often required to perform additional services and activities. This may not be captured accurately and can substantially contribute to additional losses of revenue to the business. This also includes the accurate capture of actual materials used.

The Field Force Automation solution is able to also provide a comprehensive list of additional activities, or work items that may be performed at the site. Again, this information may be entered at the time of performing the work, and will improve the capture of ad-hoc, or unplanned work.

***Improved Mobility***

Field force engineers typically collect work to be conducted at the beginning of each day. During the course of the day however, often it is necessary to alter this program of work and attend to additional (ad-hoc) jobs. In such instances, it will be necessary for the field engineer to return to base to collect details regarding the new jobs.

The Field Force Automation solution will enable the field force engineer to synchronise and download new jobs that have been assigned, thus enabling the field engineer to proceed from one job location directly to another.

***Contracts Rates Accuracy***

The scheduled contract rates provide billing rates based upon estimated time for performing work. It is often unknown how accurate the current schedule rates contract is, as such any adjustments made may actually result in a loss of revenue to the business.

By accurately capturing work performed and the time taken to perform the work, the schedule contract rates can be accurately maintained by the business, so that adjustments made do not compromise the business financially.

***Reporting and Analysis***

Under these manual or semi-automated systems, a number of staff are required to review all activities performed by the field force engineers. This is in order to generate reports for management and accounting purposes. The accuracy of the reported information is directly linked to the accuracy of the information captured and entered by field force engineers. As outlined in previous statements above, this varies considerably with planned and unplanned activities, as well difficulties in transcribing written notes into the reporting systems.

With the accurate capture of information from a Field Force Automated solution, the accuracy of reporting can be improved significantly. This will also contribute to a more accurate adjustment of the schedule contract rates by ensuring that estimates (and hence cost) for the work to be performed, are consistent with the actual work performed.

***Capture and Associated Actuals with Billing Activities***

The current practices of some organisations may require the field engineers to perform the recording of hours worked in a separate sheet; furthermore, this may be recorded quite differently to the billable activities performed. This makes it very difficult for the business to analyse the actual cost for the detailed billing activities performed, directly impacting the accuracy of the schedule contract rates and revenue capture.

The Field Force Automation solution is able to provide an integrated solution that will associate the worked hours with the billing activities. This will then ensure the collection of actual hours against billing activities, and will improve the pricing model for contract rates. Furthermore, several additional features may be included:

- Perform cross checking of dependencies, for example when the field engineer selects one particular item, the PDA can check if associated items have been selected as well.
- Additional information can be provided so that more detailed information regarding variations is captured.

### *PDA Connectivity Options*

The traditional approach of developing and enhancing corporate, or legacy systems are based on client/server architectures. In this case, the server may be a complex UNIX application hosted by some mid-range platform, whilst the client application may be developed in environments such as C++, Visual Basic, and Powerbuilder. More recently, the emergence of the Web has seen this traditional client/server based approach move towards a thin browser client model, thus overcoming device dependence of the traditional client applications and deployment problems.

By fostering the inherent benefits of web enabling corporate and consumer based legacy applications, a new and interesting paradigm is introduced; this is achieved by converting the traditional client/server application, often referred to as "fat client", to a browser based client model (thin client), that is suitably crafted for deployment to a PDA device. The novelty in the approach shown here is the intent to simulate an on-line user experience whilst the mobile worker is actually disconnected (off-line) from the network. This has a resultant effect that enables the mobile worker to fully complete their tasks, before needing to establish a mobile connection to the corporate network (and legacy system). In order to articulate this solution approach, we first review some traditional application deployment models.

### *Traditional Client-Server Applications*

The traditional "fat" client application typically comprises a number of heavily populated screens (see Figure 1), or dialogue boxes, with complex business processing logic. The dialogue boxes may contain a number of input and output fields such as list boxes, text areas, check boxes and drop down lists. To bring the interface to life, functional logic must also reside with the application so that the user is able to interact with the system. This may take the form of field validation, data formatting and transformation, and retrieval of further information from the server application with which it is connected.

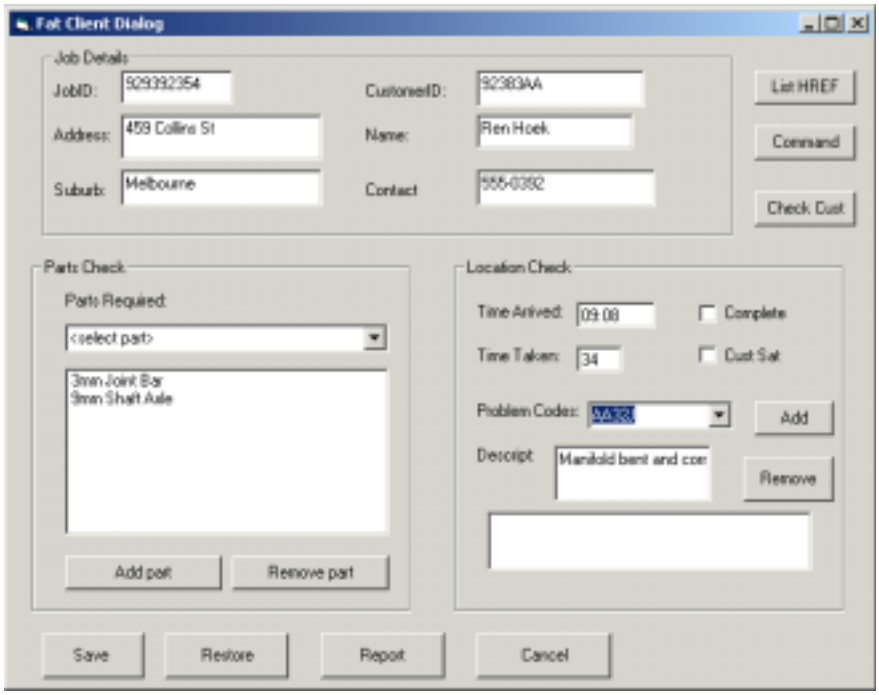



*Figure 1. Client Application User Interface*

Traditional client-server solutions are based upon the deployment of application software to the client platform. This meant that software distribution and controls are necessary to ensure that client platforms contain the appropriate software to enable connectivity back to corporate legacy systems.

The client and server applications typically communicate via a proprietary protocol (see Figure 2). Data is exchanged and the user then disconnects from the Server.

Once disconnected, the user may operate the client application utilising the data that was received from the server and also capturing any new data required of the application.

A key inhibitor of this type of application is the ability to scale to any large number of users. This was evident in the fundamental client-server design, where it is necessary to communicate over a network medium to obtain further data so that the end user is able to complete the function or activity.

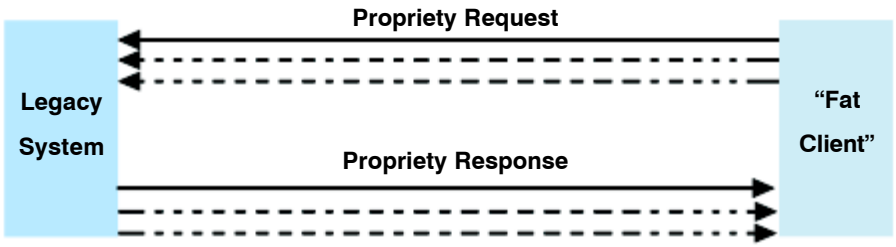



*Figure 2. Client and Server interaction*

### *Browser Based Applications*

With the advent of the Internet, the traditional client application has given way to the browser based platform (see Figure 3). In this approach no longer is it necessary to deploy application code to the platform, rather a set of instructions (HTML), describing what user interface is to comprise is forwarded to the browser. These instructions, or markup language, is interpreted by the browser for rendering to the users screen. In general, the user must enter in the appropriate data before submitting the page back for further processing. However, extensions via scripting languages such as Javascript are available to enhance further the dynamic capabilities of the HTML page.

Where further dynamic functionality is required, the ability to download a Java Applet is provided. In this sense, the Java Applet is once again a manifestation of the "fat" client. However, in this situation the Applet overcomes some of the problems of traditional client applications such as software distribution, by downloading this each time as required (note that in some cases it is necessary to deploy additional jar files to these client platforms where the Applet is of significant complexity).

A key characteristic of both the client-server and browser based solution for traditional computing environments, is that both these solutions are always connected to the networked environment. We now turn our attention to the situation where such a connected mode of operation may not be relied upon, as is typical of the mobile device user.

### *PDA Palm: Intermittent Device Gateway Connectivity Option*

Several pragmatic IBM approaches exist to deploying solutions to the PDA. The first is based upon a client-server architecture, employing what is referred to as the Intermittent Device Gateway (IDG). The second is to use a browser based architecture, and a third approach involves a combination of the client-server architecture over Internet based architectures and protocols. In each case, connectivity to the networked environment is not always assumed, in other words the PDA devices may not always be on-line. In this section we now describe the first approach and how this relates to the third approach, in the next sub-section the browser based architecture is then described.

The following solution description is detailed more fully in the IBM white paper "Extending SAP Systems to Pervasive computing devices: Enabling e-business everywhere", providing an overview of how a client-server solution may be built.

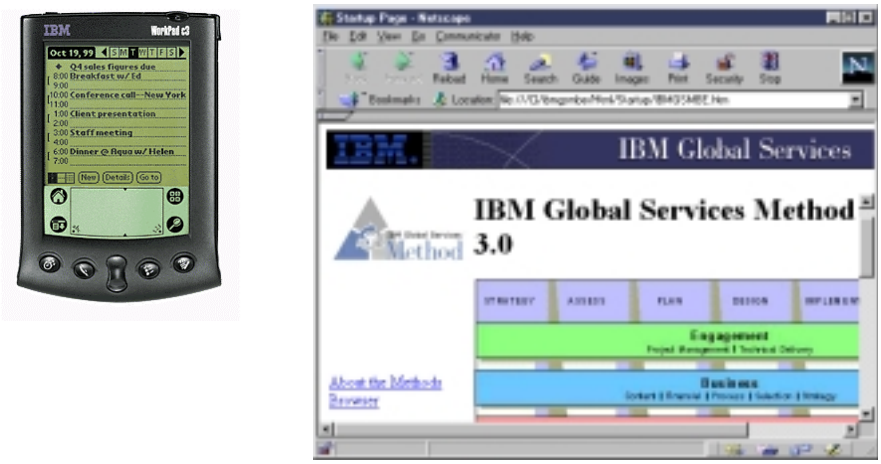



*Figure 3. Browser Application*

The Intermittent Device Gateway (IDG) is used to communicate with intermittently connected devices; for example, a Palm Computing Platform. The IDG forwards device requests to the application server in a consistent format. In this example the request format is HTTP. The IDG acts as a HTTP client, submitting the requests and waiting for the replies that are forwarded on to the intermittently connected device.

There are several architectures for the Intermittent Device Gateway that are specific to the individual needs of the applications being developed (e.g. synchronous versus asynchronous communication, data synchronisation versus transactional messaging). The following diagram illustrates the typical components used in a solution employing the Intermittent Device Gateway.

Within the Intermittent Device Gateway the following processing occurs when the user establishes a connection to transmit information.

1. The user of the client device works off-line. When the user makes a physical connection, the client device transmits packets of transaction data from the client device to the client proxy.
2. The client proxy translates these data packets into HTTP requests. It sends the request to the application servlet. As far as the application is concerned, the client proxy is just another Web browser.
3. The XML data output from the application servlet is reformatted by the transcoder, and returned to the client proxy, which assembles the responses into packets and places them on the output queue.
4. When the client device makes a request to receive data, the intermittent client device proxy services the request by sending packets to the client device. Application code on the client in turn receives the packets (via gateway connectors) and presents the data to the user.

The third approach is largely an alternative technique to delivering functionality without the dependence upon the Intermittent Device Gateway. An Intermittent Device Gateway, such as the Everyplace Synchronisation Manager, becomes essential where large volumes of data are to be moved. This is typically in Graphical Information Systems (GIS), where map images are to be delivered to the client PDA. Where the problem domain does not involve such data intensive requirements than a simpler model may be developed where the client application communicates with the server via the HTTP protocol. In both cases, a client application is resident on the PDA device.

### *Mobile Connect Gateway (Palm)*

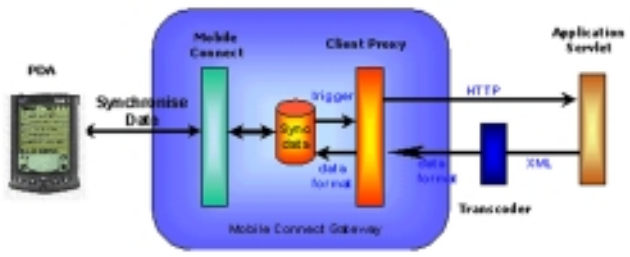



*Figure 4. Intermittent Device Gateway Model*

***PDA Browser: Simulated On-line Connectivity Option***

An interesting paradox occurs when accommodating the requirement to provide disconnected operation of the application using the PDA Client Web Browser. By convention, the HTML page is a request-response paradigm that involves perhaps several rounds of interaction between the user and the Web application server. This interaction may be augmented through the use of additional Javascript logic, however the inherent feature of submitting data via a form must be overcome (since connectivity may no longer be relied upon) in a manner that provides the user with a simulated on-line experience.

There is also the additional consideration with the deployment on a PDA browser; there is an immediate restriction on screen real estate to accommodate the same HTML page, which is usually deployed to conventional browsers. In spite of these shortcomings, an elaborate solution approach may be taken that makes use of browser based technology whilst operating in a disconnected state.

The general approach to providing such a solution is based upon segmenting a large HTML page in several many smaller virtual HTML pages, referred to as cards. The notion of a HTML card is synonymous with a card of the WAP protocol suite – a card is a small page that enables the user to interact with the system. These cards are then displayed one at a time to the user, as if interacting with a Web server, as the user migrates through the browser application. The key to enabling these HTML cards, is the use of Javascript to provide the dynamic interaction typically required of corporate applications. In general a client application incorporates a number of business rules. These rules are imposed when the user interacts with the application and also when the application communicates with the server. As such, any data and business logic necessary for operation of the application must be available on the device whilst the user is disconnected from the server. The business logic is implemented using any scripting language provided by the PDA Web Browser, either Microsoft's JScript or Netscape's JavaScript, both of which are variants of the ECMA-262 scripting standard.

The following diagram, Figure 5, illustrates the elementary ideas of this approach, where it can be seen that one large HTML page is divided into four individual areas, that are individually displayed as HTML cards. Navigation amongst the four cards is controlled by the selection of URL links or button selection.

This presented scenario leads to the following interactive steps, which are performed by the user of the browser enabled PDA device:

1. User establishes connection with Web application server, download the necessary data and business logic in one (large) HTML page, recalling that the HTML page is in fact composed of several smaller virtual HTML cards that may be displayed individually on the PDA browser.
2. The user disconnects from the mobile network, and performs the desired operations at the designated field site. The user may in fact perform several such operations whist disconnected.
3. At some time in the future, the user re-establishes a connection with the application server, via the mobile network, and up loads any captured or processed data.

Together with IBM's wireless computing platforms for deploying mobile solutions, Websphere Everyplace Server (WES) and Websphere Portal Server (WPS), corporate strength Field Force Automation solution can be developed using a browser based architecture. These specific architectural models will now be explained further.

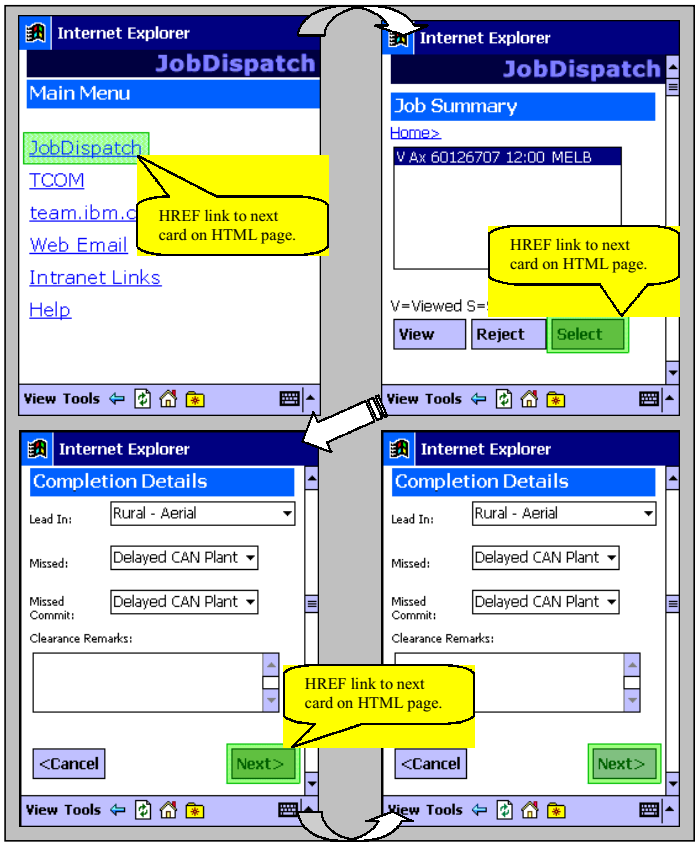



*Figure 5. Cards within one HTML Page*

### *On-line Simulation to Mobilise the Field Workforce*

In order to create a scalable solution that equips a fully mobile field workforce that number in the thousands several architectural considerations must be made. This includes device agility, off-line enablement, and responsive behaviour. These and other solution aspects may be solved by employing XML based standards approach for generating content; transcoding publisher to generate, transform and enable generated HTML cards sets; and building upon well tested scalable platforms such as the Websphere Application Server and Websphere Portal Server.

### *Architectural Overview*

The following diagram, Figure 6, provides an architectural overview of the key components of the solution. In such a solution, a key principle is to re-use business investment by ensuring that one or more legacy systems that provision field jobs to the work-force are leveraged.

In addition to mobile devices such as PDAs, mobile phones, and Laptops connected via a mobile phone, conventional PC platforms may also be used to access the content supplied by an automated work force management system. In the sections that follow, a description of how the key solution components interact is given, this is then followed by a further explanation on each of these individual components.

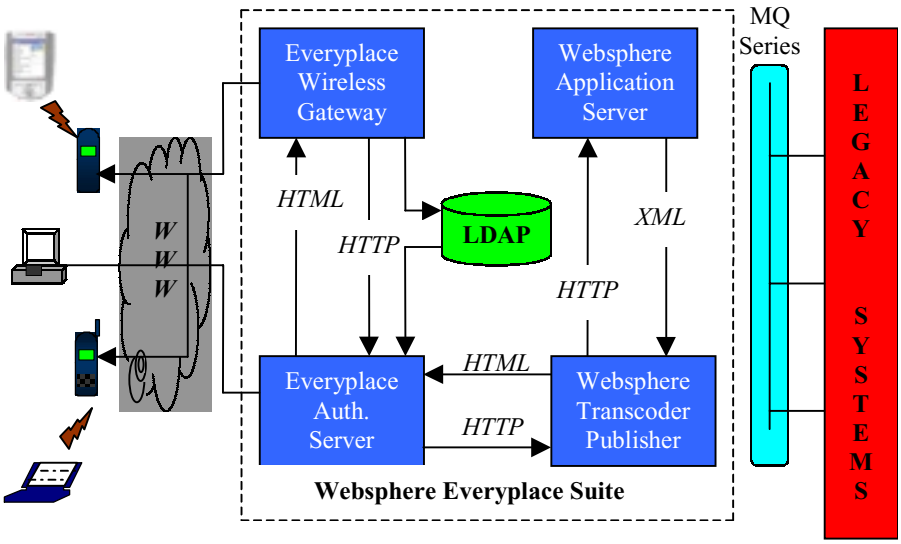



*Figure 6. Architecture Model*

***Component Interaction***

In order to initiate access to the Automated Field Force system, the user must invoke the local PDA dialler to establish a connection using the mobile phone. This is performed by running the Everyplace Wireless client that automatically invokes the dialler; note that the mobile phone actually dials a number which is attended to be a modem pool within a Remote Access Server (RAS), see Figure 7.

This connection is made to the Everyplace Wireless Gateway, where authentication and authorisation takes place, creating an active session record to a directory. Once the connection is established the user invokes the local browser, such as Pocket Internet Explorer, and accesses the URL of the Field Force Automation application. This URL access sends a HTTP request to the Authentication Server, which verifies the identity of the user, against the generated active session record in the directory, before passing the request onto the Web Application Server.

For the sake of simplicity we now view the interaction between the PDA, web application server, and existing legacy (or back-end) system. The web application server receives the HTTP request and performs the required local processing before it initiates a transaction with the legacy system. The communication with the legacy system is conducted over an asynchronous message bus provided by MQSeries, see Figure 8.

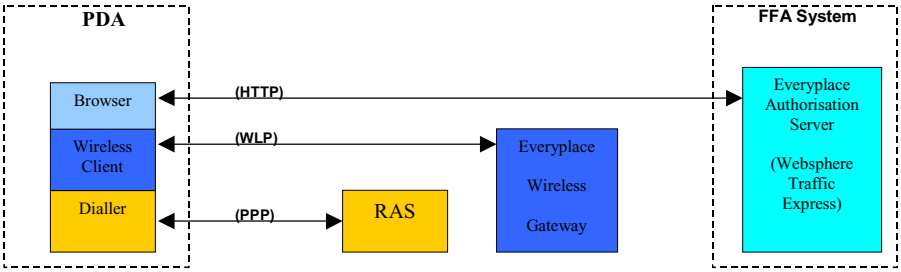



*Figure 7. Dial-up Initiation and Access*

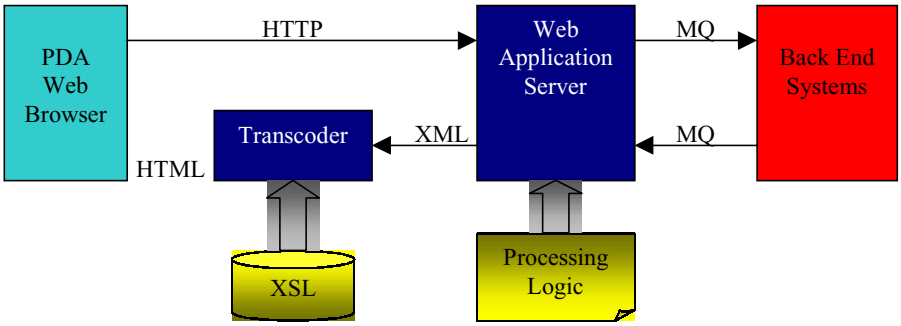



*Figure 8. Web application processing with legacy system interaction*

The legacy system conducts the required business processing on the request, and returns a response to the web application. The web application then transforms the response into the device independent XML message, and forwards this to the Transcoding Publisher for final translation into the device specific form. In the case of the browser based solution, this transcoding process also involves the addition of scripting logic so as to suitably enable the response set of HTML cards to provide users with a fully simulated on-line experience when conducting work.

The following list summarises several key technical principles of this solution:

- ECMA-262 implements the presentation and view logic consistent with the Model View Controller (MVC) design pattern.
- Additional business (Model) logic is performed at the application server, such as data manipulating and further data validation rules.
- Legacy system is made use of for the critical business functions of field force automation.
- An XML data island is employed as a tree structure to add user generated data. This is also returned back to the web application server for processing before forwarding to legacy system.
- A persistence function is provided through the use of cookies; this is also provided implicitly by opening multiple browser windows for access to various other (i.e. non FFA) business functions.

The collective interactions and related components are shown in the diagram below, Figure 9.

The key IBM solution components that have been discussed will now be viewed individually.

***Central Components of the Solution***

The browser based wireless solution has as its foundation the Websphere Everyplace Server; this can be further augmented with Websphere Portal Server. Websphere Everyplace Server is the mainstream platform for providing the necessary solution components. Where additional content delivery is required, in addition to a Field Force Automation service, Websphere Portal Server can be employed to extend the wireless solution into a full wireless portal capable of delivering multiple types of products, services, and content to its user community. This extended content can also be made available to not only the internal user community, but also the retail customers and the small to medium business customers of the business enterprise.

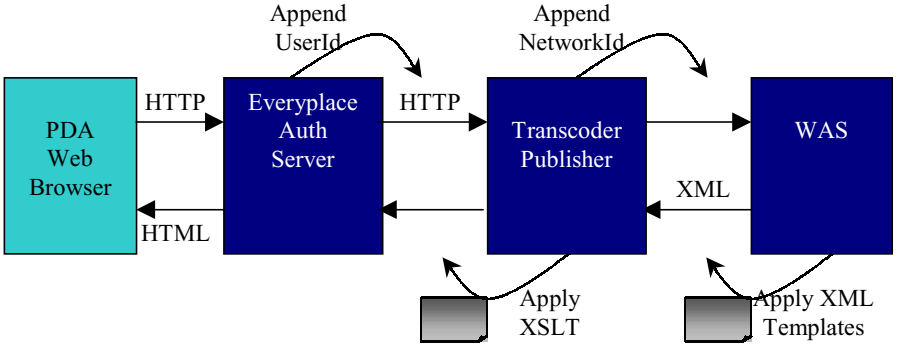



*Figure 9. End-to-End Interaction*

***Everyplace Wireless Gateway***

A key component of the WES platform, the Everyplace Wireless Gateway (EWG), provides a secure and scalable means for remote devices to establish connectivity with the Field Force system. The central function of the gateway is to establish an optimised communication link between the remote PDA device, and the application server hosting the Field Force solution. The wireless gateway can accept connections over several open protocols including IP, IPSec, WLP, or direct PPP dial up to modems attached to, or controlled by the Gateway. Where necessary, this may also be used to create a Virtual Private Network (VPN) between the users of the system and the corporate network.

The wireless gateway may also be configured to operate as a WAP gateway, accepting connections directly from mobile phones supporting the WAP protocol, and microbrowsers able to render the Wireless Mark-up Language (WML). This is in addition to its support of a number of client platforms that includes Windows, Palm's, and Windows CE. This means that as the solution is extended, use is made of the existing investment in the Field Force wireless solution.

***Everyplace Authentication Server***

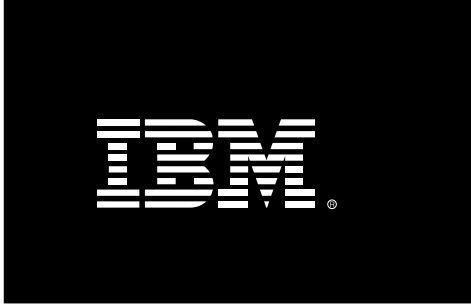
The Everyplace Authentication Server is a component of the Everyplace Server providing two key functions. The authentication server is an access point for users that are not required to establish a connection from a remote mobile device, i.e. from Personal Computers connected via the Internet, and this server also performs the security functions to authenticate users. As central node for accessing the services of a Field Force solution, since not all connections will be routed via the Wireless Gateway, the authentication server facilitates a single sign-on function.

The authentication server is actually a core sub-function that is provided by the Web Traffic Express caching proxy. As such, the server not only provides a single sign-on service, it also provisions a node from which the solution may be optimised further through its caching functions. In general the authentication server may be configured to authenticate users of the system using two approaches:

1. Authentication Proxy. Authentication is based on HTTP authentication headers. No other content server in the Everyplace domain may do its own user authentication. Users authenticated through the authentication proxy may not access content outside the Everyplace Server domain. In this mode, the Web Traffic Express must be configured as a reverse proxy.
2. Transparent Authentication Proxy. Authentication is based on HTTP proxy authentication headers. Content servers in the Everyplace Server domain may do their own user authentication. The transparent authentication proxy allows users to access material outside the Everyplace Server domain (for example, the Internet).

***Websphere Transcoding Publisher***

The Websphere Transcoding Publisher (WTP) is a component within the Websphere Everyplace Server. The transcoder is an integral part of the solution, providing a mechanism for converting the XML content created by the application server, into the device specific markup language – in the case of the Field Automation solution this is HTML. This conversion is performed under the control of an XSL style sheet, containing XSLT logic, all of which are based upon open standards. A style sheet is typically developed for each device that is intended to receive the content. As such, style sheets are developed to model the specific characteristics of each device that is intended to render the content supplied, facilitating the separation of content and device. Through this unique separation of content and the target device, new devices may be added to the portfolio of portable devices capable of receiving field force instructions from the Field Force solution. Furthermore, the application business logic need only be written once, with the view logic developed for each target device.



Due to the need to simulate an on-line experience, whilst users are disconnected from the field force back end system and application servers, it was noted that additional logic is also inserted into the output HTML stream for each device. This logic, typically Javascript, is only required to be written once and may be re-used by each composed XSL style sheet.

### *Websphere Application Server and VisualAge*

IBM's WebSphere Application Server (WAS), is the application server used to manage the web application servers shown in the solutions discussed above. A web application server is necessary to provide a productive, robust, scalable, and reliable platform for building large-scale web and wireless e-business solutions. Websphere builds upon the J2EE standard and applies industry standard protocols such as the HTTP protocol and XML standard to interact with the Transcoding Publisher.

The fundamental interactions with WebSphere involve the processing of incoming HTTP requests, sending and receiving MQ Series messages to and from back-end legacy systems, and outputting device independent XML content. Development of the Java logic of the application server is performed using the fully integrated environment that VisualAge provides.

The Pocket IE browser is supported on the Windows CE platform, however several restrictions apply to its use. This is in terms of the available screen size, the processing power of the PDA device, and downlink protocol speed to accept data from the application server. This means that the final application must be carefully crafted in order to provide users with a responsive solution, which is user friendly, and provides sufficient functionality to enable the mobile work force whilst performing activities disconnected from the network.

An implementation based upon the use of HTML cards, is able to provide the required field force services, whilst not overburdening the PDA device with a large application client.

### *Summary*

Two IBM approaches exist to build a wireless solution for Field Force Automation. This first involves the use of an Intermittent Device Gateway, and is more extensively explored in the IBM white paper "Extending SAP Systems to Pervasive computing devices: Enabling e-business everywhere". The second approach is based upon a Browser based architecture, and is the solution that has been treated in detail by this paper. In both approaches the key requirement of enabling a disconnected mode of operation must be supported. A browser based architecture however, provides a very flexible solution that enables an organisation to maximise its investment in Internet technologies and standards. This offers a scalable solution, that overcomes the problems associated with the deployment and maintenance of large application clients to mobile devices. Furthermore, the use of the technology and approaches described here, extend well beyond the boundary of Telecommunications and Utilities based organisations requiring a solution to their Field Force management problems. Moreover, the solutions can be readily applied to other forms of mobile workforces such as Sales Force Automation.

### *For more Information*

To learn more about IBM Global Services and Pervasive Computing, contact your IBM sales representative or visit:

Global Services Website at: www.**ibm.com**/services
Pervasive Computing Website at: www.**ibm.com**/pvc/tech

## APPENDIX I. Legal

### *General Considerations*

THE DATA IN THIS WHITE PAPER ARE PROVIDED ON AN "AS IS" BASIS FOR INFORMATION PURPOSES ONLY, AND MAY CONTAIN TECHNICAL INACCURACIES. ALL LIABILITIES FOR DAMAGES, DIRECT AND INDIRECT, ARE HEREBY WAIVED, AND ALL WARRANTIES ARE HEREBY DISCLAIMED, INCLUDING WITHOUT LIMITATION, THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, AND NON-INFRINGEMENT, EXCEPT AS CANNOT BE EXCLUDED BY THE TRADE PRACTICES ACT.

All statements regarding IBM's future direction or intent are subject to change or withdrawal without notice.

This white paper may contain references or cross references to IBM products, programs and services that are not announced or available in your country. Such references do not imply that IBM intends to announce such products, programs or services in your country. Consult your local IBM business contact for information regarding the products, programs and services which may be available to you.

Any reference to an IBM product is not intended to imply that only an IBM product may be used.

### *Trademarks*

The e-business logo, IBM, MQSeries, DB2, WorkPad, WebSphere, CorePoint, VisualAge, and Websphere Everyplace are trademarks or registered trademarks of IBM Business Machines Corporation in the United States, or other countries, or both.

Java and all Java-based trademarks and logos are trademarks or registered trademarks of Sun Microsystems Inc. in the United States, or other countries, or both.

Microsoft, Internet Explorer, Pocket IE, Windows, Windows CE, Windows NT and the Windows logo are trademarks of Microsoft Corporation in the United States, or other countries, or both.

Other company, product and service names may be trademarks or service marks of others.

[1] Reference. Extending SAP Systems to Pervasive computing devices: Enabling e-business everywhere, Rudi Jetzelsperger and Clive Gee, International Business Machines (IBM), 1999.

[2] International Data Corporation (IDC), 1999.