Roll Your Own AVL using ArcGIS Server

A Local Government's Experience

By

B. R. Carson

County of Henrico, Va
IT Department

May 23, 2012

Abstract

Submitted Title: Roll Your Own AVL using ArcGIS Server

Submitted Abstract:

Henrico County's Cad24x7 system is a modern service-oriented high-availability 9-1-1 dispatch application. It was designed and developed in-house by the County IT Department. The County has added an automatic vehicle location (AVL) capability to the system using ArcGIS Server. AVL provides: GPS data capture from Police, Fire and EMS units; browser-based tiled-map for unit location visualization; Network Analyst based closest unit recommendations and unit routing features which are integrated into the existing dispatch and mobile data applications. The presentation will review: system architectures for Cad24x7 and the enterprise GIS; the role our ESRI ELA played in helping us to realize the vision; how the ArcGIS Server SOAP and Rest API's were used to implement our system. This project illustrates how sophisticated unit tracking, routing, and spatial closest-unit recommendations are within the reach of the average application developer by leveraging ESRI based services to do the heavy lifting.

Introduction

The County.

Adjacent to the north and east of Richmond, the County of Henrico, Virginia (pronounced hen-rye-co) covers 244 square miles with a population of approximately 310,000 residents. In Virginia the counties and cities are separate political entities. As a result, Henrico maintains its own emergency services totally apart from the City of Richmond, including: police, fire, EMS and emergency communications. The County's Division of Police has 605 sworn officers deployed from two precinct locations. The Division of Fire staffs 20 fire stations which provide fire suppression service with 14 of these also providing EMS service. Four volunteer rescue squads also provide EMS services to the County. All are provided with GPS equipped mobile data computers. The emergency communications function for the County is managed by the Division of Police and includes a 30 position E-9-1-1 communications center.

The emergency communications center uses a computer aided dispatch (CAD) system to aid it in managing the nearly 390,000 calls for service and unit initiated activities it receives each year. This system was developed in house by the County IT Department which has provided the CAD system for the County for over 30 years. The current system, called Cad24x7, was implemented in mid-2008 and is a modern service-oriented high-availability 9-1-1 dispatch application.

The Problem.

One of the design goals of the CAD system was to incorporate the features commonly provided by automated vehicle location (**AVL**) applications, principally: GPS based unit tracking; GIS based visualization (mapping) of these units; proximity based unit recommendations for call assignments; and ancillary functions such as routing and eventually deployment modeling. However, for a couple of reasons these goals were not realizable back in June 2008 when the Cad24x7 system first came online. Firstly, our enterprise GIS was not architected in a manner that made it reasonable to allow the needed system integrations to happen. Secondly, the IT software development staff did not have the GIS expertise it would have taken at the time to implement such features. This article relates how changing our GIS architecture to incorporate ArcGIS server provided the means to leverage the ESRI services needed to make this project happen.

The AVL Problem Set.

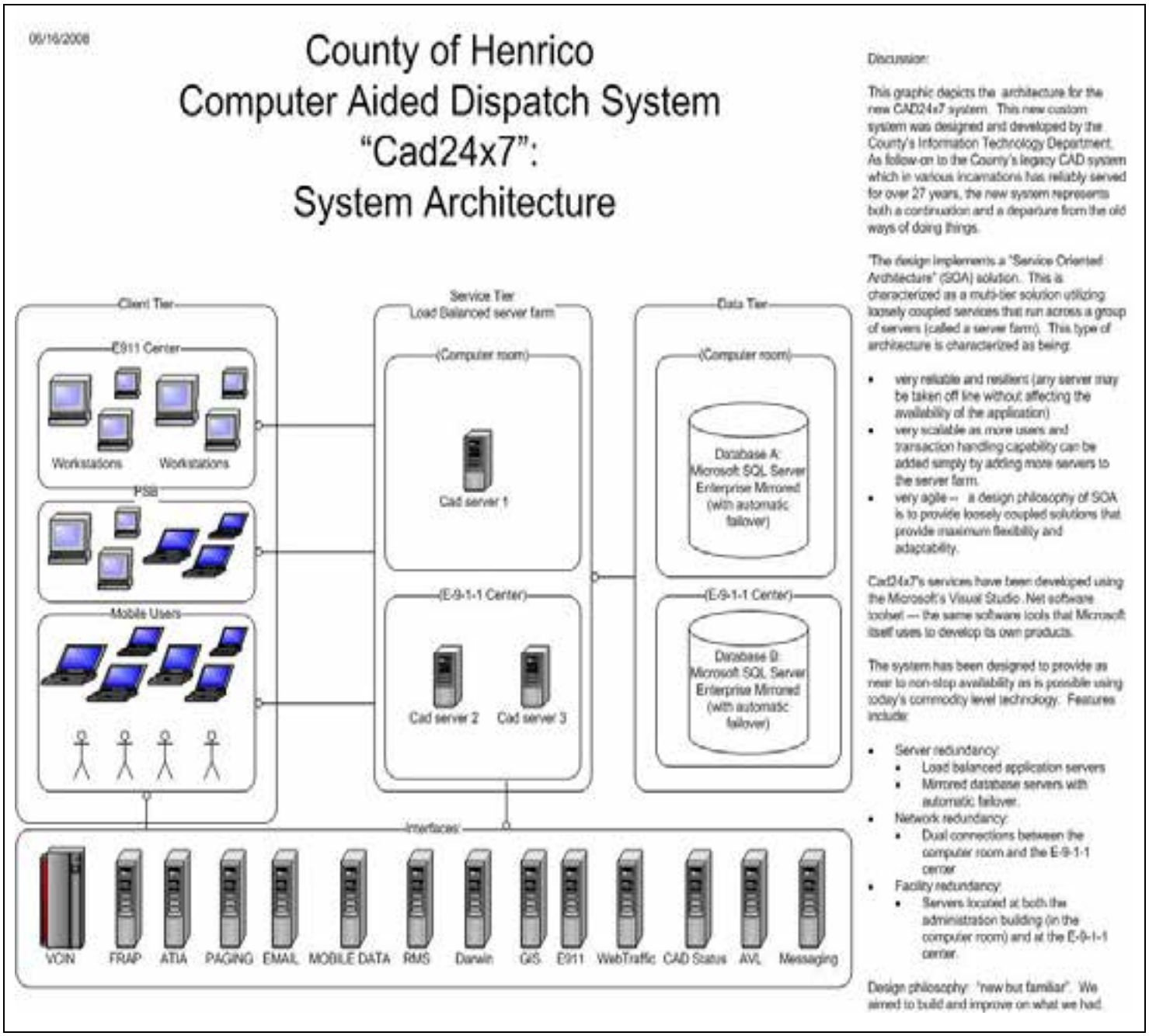
From the local government perspective AVL has always been one of those applications you purchase from a vendor because it involved technologies that were largely out of reach for the local IT department. However, if you decompose an AVL application into its component programming tasks and evaluate these tasks versus the skill set of the typical application developer you'll see that most are relatively mundane IT software development tasks: reading data from a serial port, saving it to a database, writing business logic, creating web services. However, there are a couple of items on the list that are way out of reach of most programmers: mapping and routing, for instance. This is where ArcGIS server comes in. It provides the means to readily create the mapping and routing services needed to implement an AVL system. It provides industry standard API's, like REST and SOAP, to access these services that allow the local application developer to "stand on the shoulders of giants" and leverage powerful software which would normally be far beyond his/her capability to develop.

System Architectures

Cad System Architecture.

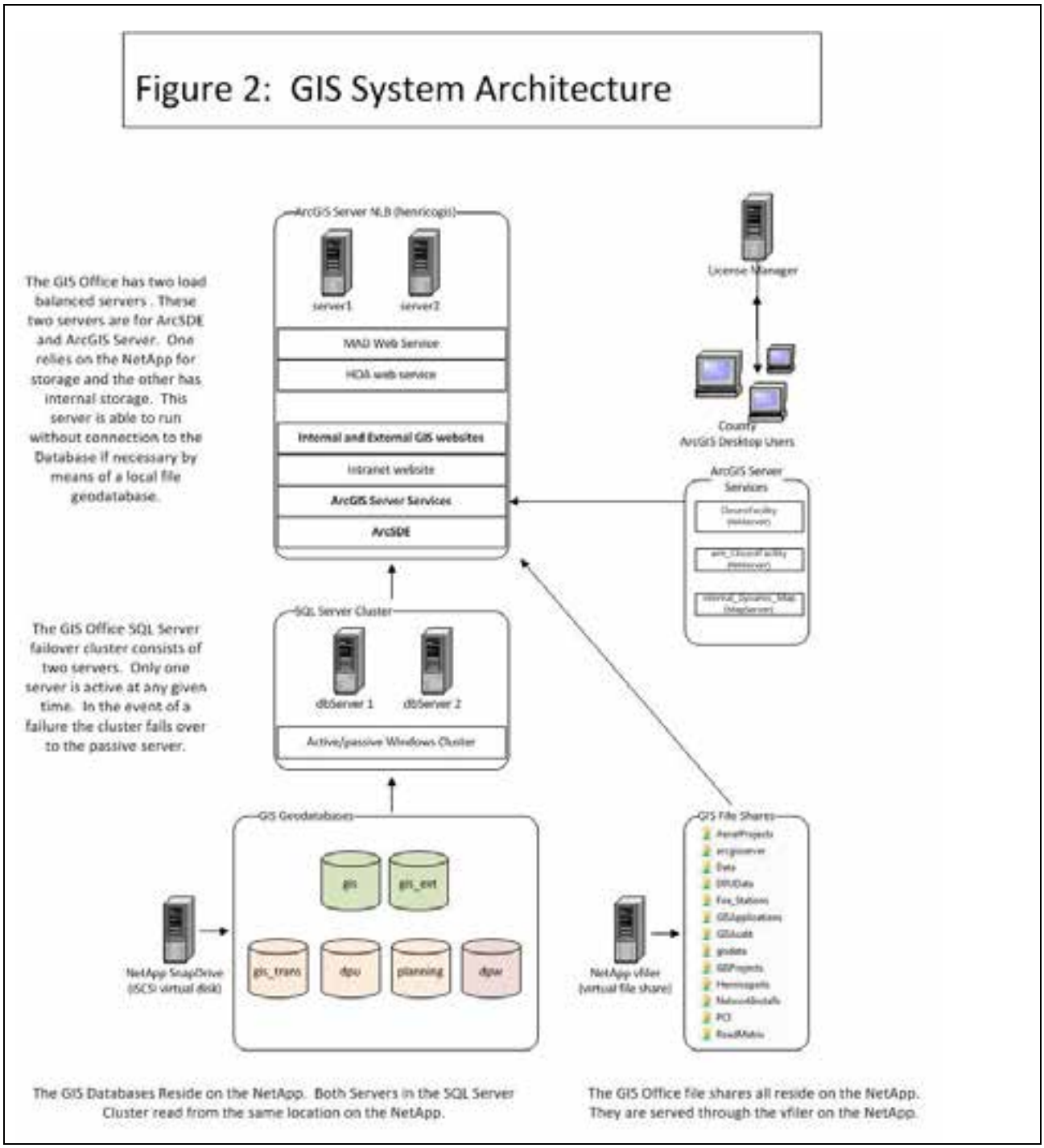
Henrico County's CAD system, which we call Cad24x7, is a high availability service oriented application which was developed in house and was implemented in 2008. The system was written in Microsoft Visual Studio.net (VB.net) using MS Sql Server 2005 as the database. The application is deployed over five servers, three of which are network load balanced (NLB) application servers with two being mirrored database servers. There are two application servers and one database server sited at the emergency communications center and one application server and the other database server at the County computer center which is approximately 1 mile away. The idea is to provide a facility independent business continuity capability. Figure 1 provides a graphical view of the system architecture.

Figure 1.



GIS System Architecture.

Our GIS architecture also provides a high availability solution with ArcGIS server being provided over a pair of NLB servers and the SQL Server 2008 database being hosted on an active/passive Microsoft Cluster server. At present these servers are located solely in the County's computer room. We plan to relocate one of the NLB servers to the emergency communications center later this year so as to provide some measure of facility independence. Figure 2 provides a graphical view of the system's architecture.



How the ELA helped us to realize our vision.

About the same time as we were planning our new GIS architecture we began looking into obtaining an Enterprise License Agreement (ELA) with ESRI. The ELA made it possible for us to deploy the high availability architecture for the GIS system which was needed for us to meet the service level requirements of Cad24x7 system. It also enabled us to get the right toolsets into the hands of the software developers in the IT department some of whom had not before been traditional GIS users. With the ELA the focus became more about architecting the solution required to solve the problem with less immediate concern on the marginal licensing cost issues. In the end this was beneficial as some of our initial plans actually involved a larger licensing footprint than we ultimately needed.

The Henrico County AVL system

The AVL system is a subsystem of the Cad24x7 system. As such it stores its GPS readings in the Cad24x7 system's database. It does, however, have several distinct parts and pieces all its own. It consists of the following:

1. **A GPS collection service**. The GPS data collection component is implemented as a windows service that runs on the mobile data computer (MDC) in each police, fire and EMS vehicle. This service reads the NMEA spill from the MDC's onboard GPS receiver once a second. It records a local copy of the RMC sentence and reports data back to the central CAD system every 10 seconds when the vehicle is in motion or every 30 seconds when it is stationary. Data submitted include: unit number, ip-address, latitude, longitude, heading, speed, common location, date time stamp, number of satellites in view, type of GPS reading (2d vs 3d) and GPS status (to indicate errors). This data is sent up to a web service over a Verizon 3g wireless broadband link. Once received the messages are put on a message queue for post processing during which the latitude/longitude coordinates are re-projected into both NADA1983VaStatePlane and Web Mecator projections. This data is then stored on the CAD24x7 database into two tables. One table contains the most recent GPS reading for each MDC; the 2nd table is a GPS history table. We maintain 90 days of GPS readings on this table to support replay functions, etc. Current volumes average 500-600,000 GPS readings per day. Thus the GPS history table contains approximately 50 million rows. The advantage of maintaining the most recent GPS reading in a separate table is that the table is small (one reading per MDT) and therefore fast and easy to query. This makes it ideal for use with the high performance requirements of a CAD system. Also the advantage of re-projecting the coordinates once and storing the result on the database is that we are mainly consuming the data in either Web Mecator or State Plane format and may read and use/display each entry several times without the need to convert the coordinates. A special note, GPS data collection has proven to be the most challenging part of the entire project –

especially with the larger apparatus types like fire engines and ladder trucks.

2. **Visualization of the GPS data**. One of the goals of the AVL design was to provide an intelligent visualization of where the police, fire and EMS units were. Towards this end we created a web map project using ArcGIS server's Silverlight API which provided a beautiful tiled base map as a canvas upon which we could display information from our CAD system regarding the various units, their activities and their whereabouts. Figure 3 provides a screenshot of the map. The CAD unit data is based upon information exposed by the CAD system as a web service. This data is cached by the CAD web server with a life of 10 seconds. So most requests for information receive a reply from the cache. When the cache expires the next call to the web service causes a fresh query to be issued against the CAD database and re-cached. This enables this application to scale well and support hundreds of concurrent users with very little overhead. The tiled map is similarly efficient and once displayed seems to result in a call to our GIS servers only if the map is panned or zoomed.

   Each police/fire/EMS unit displayed on the map is rendered as an unit icon which displays a color coded id tag immediately above the icon. (Figure 3). The reddish icons represent units which are currently actively assigned to an activity. The greenish icons represent units which are currently available. For both reddish and greenish icons the darker shades indicate a unit which is stopped and the lighter shades indicate units in motion. By "mousing over" a unit's icon the user causes a terse report of the unit's current status to display in the upper right corner of the map. A double click on a unit will actually query the unit's current or most recent activity from the CAD system and display it in a new window.

   The menu icon, to left, expands to provide a set of menu options. The VIEW menu provides options and filters for selecting what the map will display.
   The AERIALS menu selection is scale dependent and only displays when the map is sufficiently zoomed in. It too causes a new window to display. In this case it launches a web viewer for our Pictometry passing it extent boundaries on the URL. Once the user has finished with this window they simply close it.
   The NOT REPORTING menu option displays a list of units not currently reporting their GPS positions with an indication of why they are not reporting. This may be because they are at a known location and have exited the vehicle; or because they are in the bay of the Fire station; or because they are in a location that provides inadequate reception.

   The REPLAY menu item provides authorized users the means to replay unit motions over time or to export data to a spreadsheet or .kml file. The ROUTING menu item provides routed directions from a unit's current location to a specified call for service or to the closest fire hydrants.

   The HELP menu items provides some guidance in map usage and access to a legend popup, a mobile toolbar, an about page and a contact-us email dialogue.

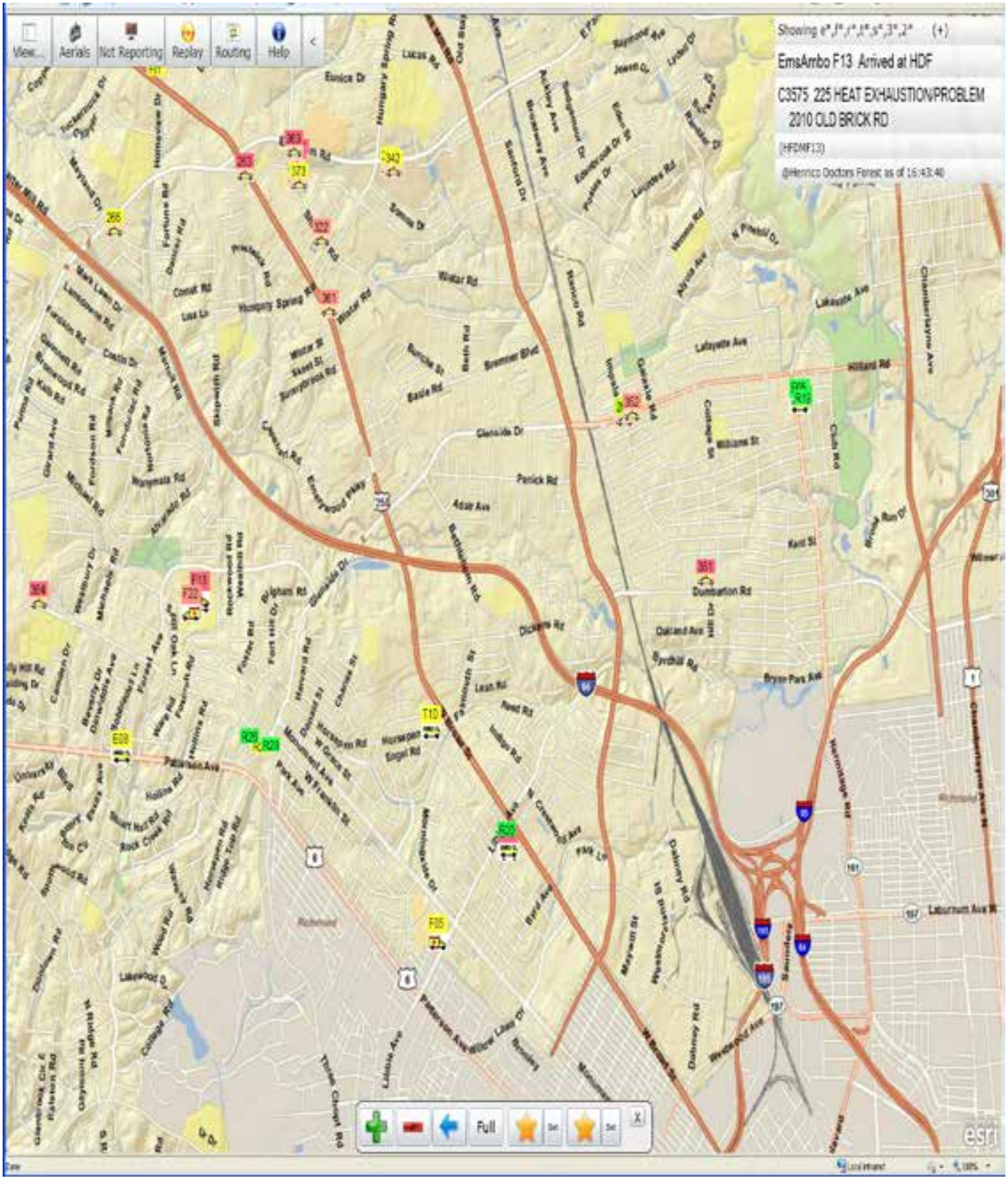

View...
Aerials
Not Reporting
Replay
Routing
Help
EmsAmbo F13 Arrived at HDF
C3575 225 HEAT EXHAUSTION/PROBLEM
2010 OLD BRICK RD
(HFDMF13)
Full
esri

Figure 3

3. **Proximity based unit recommendations**. One of the main design goals of the Cad24x7 system was to provide closest unit recommendations for emergency response. ArcGIS server makes this possible. We determined that we could leverage the closest facility solver feature of the Network Analyst service to accomplish this. Fortunately the County had previously created a networkable streets dataset based upon our street centerline layer so we were able to implement a NAServer/ClosestFacility service.

The way to determine the closest units to a call for service using the Closest Facity solver is that you treat the units' locations at a point in time as the facility locations and you treat the location of the 9-1-1 call as the incident location. Thus the most recent GPS reading from each of the units capable of responding are passed to the solver as facility locations and the location of the call for service is passed over as the incident location. The Closest Facility solver provides a lot of options. For our purposes we turned off many of the more expensive optional return items. For instance, we use the following settings:

```
Directions = False
Facilities = False
Incidents = False
Map = False
CFRouteGeometries = False
CFRoutes = True
Direction =
 henricoGisSoap.esriNATravelDirection.esriNATravelDirectionFromFacility
```

This basically gives us a data only exchange with the GIS. ArcGIS Server exposes the Closest Facility service via both a REST and SOAP API. Our initial tests with the REST API's showed that we could only send in locations for a few units before we exceeded maximum URL lengths so we used the SOAP API simply because it was easier to pass a lot of parameters in that way. The SOAP API returns an ESRI NAServerClosestFacilityResults object. This is essentially a container that holds the results of the solver call which is expressed in variety of ways. In our case we were most interested in the **cfRoutes** property of this object which returns an ESRI record set object. It is possible from the information returned in the **cfRoutes** record set to extract information identifying each unit and it's distance, ETA and rank-per-eta vis a vis the call for service location. This exactly the kind information we need to determine which units should respond to the call for service.

<u>Technical details</u>**.** Firstly we encapsulated the code all into a single class which we called ClosestUnitsSoap. This exposes only a few public methods. The one we primarily use is simply called GetRecommendedUnits(). So from the Cad24x7 system's point of view implementing this function is very simple: create a list of candidate units and pass this in along with information on the call for service. The units list is used to populate the NaServerParams.Facilities collection; the call for service information to populate the NaServerParams.Incidents collection. These two items are the principal inputs into the NaServer.solve method which does the actual calculations. Note that we do some pre-processing to limit the size of the facilities collection since it is not uncommon to have information on fifty (50) or more candidate units. Basically, we use a simple distance formula to limit the number of units we pass into the NAServer.ClosestFacility service to 10. This is a feasible step for us as we are working in a state plane (NAD83) coordinate system. This helps us keep the response time reasonable: it is typically three (3) seconds or less (1.9 seconds on average). We set a five (5) second time out on the call after which time the Cad24x7 system simply returns a static recommendation to the dispatcher. The Solve method returns an object of type NaServerClosestFacilityResults. This contains an ESRI record set object named CFRoutes. The ESRI record set object is essentially a collection of name-value pairs and as such is relatively self documenting which was very helpful as I was unable to find any documentation to aid in the interpretation of

this data. What I did, instead, was to examine the returned record set in a quick watch window in my Visual Studio IDE and to determine which items I would need to use by inspection. This sort of approach makes for very tender code – meaning updates to ArcGIS Server need to be tested to insure the content of the record set hasn't changed. While this is not an ideal solution, it is a workable one and so far one that hasn't created any problems for us.

The end result of the call to the GetRecommendedUnits() method is that an ordered list of candidate units is returned. The list is ordered, ascending, by estimated time of arrival (ETA). Both ETA and distance values are returned along with the original unit oriented data. This information is then used to formulate a recommended set of units to respond to the call for service.

Results. Response time statistics indicate a very favorable outcome from our spatial recommended-units methodology. The data showed, on average, 15% improvement in first unit response time. Thus, this one change meant that units were often arriving on scene at a call one to two minutes sooner than before. Interestingly, the percentage of responses that occurred within the critical first five minutes of a call increased from less than 50% to 65% (a 30% improvement).

Caveats. In order to have a fully functional recommended units capability you need to have a fallback plan because spatial recommendations are not a panacea for a number of reasons:

- We get about a 99% success rate on our ClosestFacility service calls. Most failures are because either the incident location or too may of the unit locations cannot be located to the roadway network – so the service reports back a "no facilities" error or a "no incidents" error; or because of a timeout; or because the GIS servers were not available;

- Police, Fire and EMS units do not always report their GPS locations infallibly. We don't use any GPS reading over 2 minutes old.

- We only use spatially based recommendations for calls which require an emergency response (e.g., code 3: lights & siren). We use a static, turf-based, recommendation for the remainder of calls, which are by far the majority. This is mostly a matter of GIS server resource availability – we do an average of 215 spatial recommendations a day which covers approximately 90% of our EMS calls, 67% of our Fire calls and 6% of Police calls. Average turn around time on a ClosestFacility service call is 1.9 seconds. There are a lot of benefits to be gained from expanding this service to include some non-emergency calls. However, to include them all would increase the demands on our ClosestFacility service by approximately 500%. We are approaching this cautiously as we want to insure that our highest priority calls for service continue to receive the best service possible.

- Not all calls for service are spatially locatable (mutual aid calls, etc.) and obviously such a call is not a candidate for a spatially based unit recommendation.

The point is that an efficient and well planned static unit recommendation process is an essential to supplement spatially based unit recommendations. It provides a fall back if there is a failure in the spatial recommendation and can be used to provide unit recommendations for lower priority calls which do not require exigent response. Our static recommendations process has been in use for over 30 years and is exceptionally quick (average response time 30ms). It is a turf based process which uses small jigsaw puzzle like pieces of the County called SRA's to host response lists

of the closest stations or zones. We recommend the units assigned to work those stations or zones. As with any static unit recommendation scheme it is dumb relative to a unit's actual location. It is smart in other ways, though, and considers unit availability and assignments in making its recommendations. Because it is so inexpensive, we always run a static unit recommendation even when we intend to use a spatially based recommendation. This enables us to backfill an incomplete recommendation from the static one or to deliver the static result to the user if there is a problem with the spatial recommendation. The dispatcher is typically totally unaware of what occurs behind the scenes. Either way the system delivers the best unit recommendation it is capable of providing at the time.

Post Processing Recommendations. It should be noted that the manner in which spatial recommendations are implemented allows for them to be post-processed before the information is returned to the user. This is essential because it allows for the following things to occur:

- Application of rule sets, for instance:
    - Ensure that the correct service level (ALS/BLS) is provided for on the call;
    - Prefer fire engines to fire trucks on EMS calls, all things being equal;
    - Use the least recently used ambulance when there are more than one available at a given "location" (which we define as within a given ETA window);
    - Send two engines on interstate highway calls (one down each side of the interstate);
    - Supply a tanker truck for calls in areas where there are no fire hydrants.
- Ensure that units don't get overlooked because their GPS is not currently reporting.
- Evaluate if a complete recommendation was achieved and if not backfill from the static recommendation.

Summation

ArcGIS server has enabled us to write our own automated vehicle location (AVL) capability. It has provided a service oriented means of incorporating spatial intelligence into our computer aided dispatch application. Prior to the existence of ArcGIS server this capability was not accessible on an industrial strength server to server basis to the average application developer. It is now accessible via industry standard APIs which are familiar to IT oriented software developers. The fact that our County acquired an ESRI enterprise license agreement (ELA) enabled us to accomplish this project. Firstly, we could afford to establish a high-availability enterprise service-oriented architecture for our GIS. Secondly, it enabled our developers to gain more familiarity with GIS concepts and possibilities and supplied them with the toolsets they needed to effectively develop applications that incorporate the GIS oriented services into our application. The message, if this paper has one to offer, is that if we can do this you can do this. A computer aided dispatch system is not rocket science. Nor with the help of ArcGIS server is an AVL system.