

10<sup>th</sup> International Command and Control Research and Technology Symposium  
The Future of C2

**SHARED USER INTERFACES FOR DYNAMIC COLLABORATION IN NETWORK-  
CENTRIC COMMAND CENTERS**

*Topic: C2 Architecture or Human Factors Engineering*

**Sandeep Mulgund, Ph.D.**  
The MITRE Corporation  
202 Burlington Road M/S M315  
Bedford, MA 01730  
Voice 781-271-6207/Fax 781-271-2964  
[smulgund@mitre.org](mailto:smulgund@mitre.org)

**Abigail Travis**  
The MITRE Corporation  
202 Burlington Rd, M/S M325  
Bedford, MA 01730  
781-271-6988 / 781-271-2964  
[abigail@mitre.org](mailto:abigail@mitre.org)

**John Standard**  
202 Burlington Rd, M/S M325  
Bedford, MA 01730  
781-271-5235 / 781-271-2964  
[jks@mitre.org](mailto:jks@mitre.org)

**C. Don Means**  
The MITRE Corporation  
202 Burlington Rd, M/S M325  
Bedford, MA 01730  
781-271-8509 / 781-271-2964  
[donmeans@mitre.org](mailto:donmeans@mitre.org)

**Aaron Burgman**  
The MITRE Corporation  
202 Burlington Rd, M/S M325  
Bedford, MA 01730  
781-271-3195 / 781-271-2964  
[aburgman@mitre.org](mailto:aburgman@mitre.org)

## Abstract

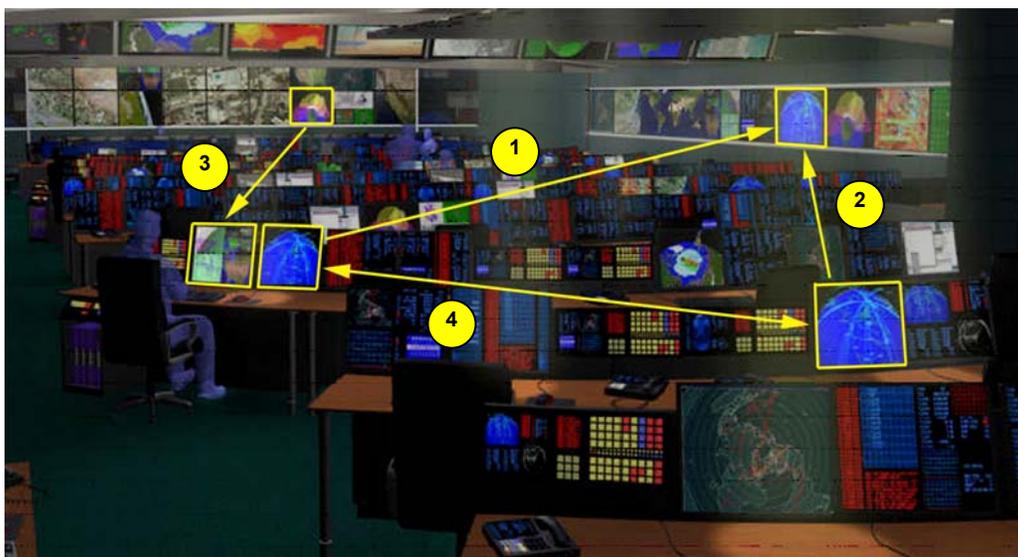
Modern military command centers are awash in complex display technology, ranging in scale from handheld computers to room-spanning display walls. Network-centric information technologies make it possible for these systems to plug into a vast web of knowledge. Yet it remains a challenge to develop effective decision support and human/computer interaction concepts that simplify this complexity and make it possible for the occupants of a command center to share mission information between systems and displays effortlessly at a semantic level. The objective of the work described in this paper was to explore new concepts for intuitive command center information sharing using collaborative publish-and-subscribe mechanisms that link together command center applications and data across multiple displays. The concepts developed through this work illustrate potential implementation architectures and human/computer interaction approaches for network-centric decision support.

## 1. Introduction

Modern military command centers are awash in complex display technology, ranging in scale from personal desktop displays to room-spanning knowledge walls. In the future this may include portable devices such as tablet PCs, personal digital assistants, or wearable computers. Network-centric information technologies make it possible for these systems to plug into a vast web of knowledge. Yet it remains a challenge to develop effective decision support and human/computer interaction concepts that simplify this complexity and make it possible for the users of a command center to share mission information between systems and displays effortlessly at a semantic level. When a need arises to share the visualizations, analyses, or other representations developed on one platform with others, standard practice is to direct the video output from that platform to a public display. Technologies such as video over IP make it possible to share such views using a data network rather than video cabling, but even so, *pixels* are shared, not *information*. This approach is limiting in that user interfaces that are optimized for desktop use may not be effective presentation mechanisms when viewed on a large-wall display, and the mirrored video image is just that – an image. There is no straightforward, standardized way for members of an interested community to interact with it, annotate it, resolve conflicting viewpoints, or bring it to their local displays for more detailed inspection. Providing such capabilities to educated but non-technical users in a command center, who are not interested in the details of distributed computing infrastructures, poses a considerable challenge.

As an example, Figure 1 illustrates the types of information sharing that may take place within a command center. In this illustration the categories of information sharing are distinguished by the degree of privacy associated with the source and destination displays. Flow (1) is the posting of information from desktop workstation displays to public displays for consumption by the broader community in the command center. Flow (2) is the visual integration of multiple workstation views onto a single view (perhaps geo-registered on a single map) on a public display. Flow (3) is the notion of drilldown: if a consumer in the command center identifies an interesting piece of information on a public display, how its underlying details be examined on a personal display? Examining such details on a personal display may be preferable for two reasons: it won't interfere with anyone else's view on the public display, and it may be easier to digest detailed textual information at a comfortable reading distance. Finally, flow (4) is the notion of peer-to-peer view sharing: how can operators at nearby workstations exchange the mission views that they are developing for mutual benefit? Note that all of these interactions here

are being performed among displays within one physical command center: each of them may also take place between command centers, introducing even more technical and human/computer interaction (HCI) challenges.



**Figure 1: Examples of Command Center Information Sharing**

It is worth exploring in greater detail what exactly is meant by *sharing* information, as the operational and implementation implications are significant. For each of the transactions illustrated above, there are at least three ways to move information from origin to destination:

- **Pixels:** The screen output of one system is sent to another, often routed through video switching equipment. While straightforward to implement, it is limited in that the view cannot be reformatted for different target devices or otherwise modified, and that multiple views cannot be integrated. The receiver cannot “drill down” into the data to understand its underlying details, and is at the mercy of the person operating the source system for what is displayed. It was our hypothesis that next-generation command centers must move beyond the pixel-sharing method to achieve effective, data-rich collaboration.
- **Data:** This is a thrust of much current work in machine-to-machine integration and the development of integrated operational pictures. The information shared between systems is a semantic representation of what is being shown onscreen; e.g., an XML document representing threat data. Benefits of this approach over pixel sharing are that the receiver can integrate multiple data views, and the views can be reformatted for different target devices. This does require that the receiver be able to understand the data format and be able to reconstruct a pixel view thereof. There are also challenges associated with transporting such data across security boundaries.
- **Recipe:** The final form of data sharing is the “recipe” level; i.e., the set of database queries or filters that are used to construct a semantic data view. These are the *instructions* used to build the data level representation. In addition to the benefits of data-level sharing, recipe sharing also makes it possible for the receiver to examine the recipe and fine-tune it to his/her needs. This necessitates that the receiver be able to reconstruct the data view from the query/recipe, meaning that access is required to the original data sources.

Current operational practice often relies on pixel sharing, which is a fairly shallow form of collaboration. Moving to data and recipe sharing makes possible an increasingly semantic, knowledge-rich form of collaboration.

Much current research and development focuses on the second level, but it is a hypothesis of this research that the exploration of “recipe” sharing may make possible an even richer collaboration, by allowing each team member to have a better understanding of the motives and assumptions used by their colleagues to construct a shared picture of the mission environment. It is also potentially more bandwidth-efficient: a database query for tactical data will generally take fewer bytes to describe than its results. When sharing a recipe it is assumed that the receiver has the ability to reconstruct the data view based on that recipe. Thus in developing operationally valid concepts for such information-sharing it will be necessary to define fallback strategies for when a higher semantic level is unavailable.

Table 1 summarizes the benefits and limitations of the three aforementioned approaches to information-sharing. As we move down the table, we believe that there is a conceptual progression from renderings (pixels) to descriptions (data) to questions (recipes). Each successive level offers greater opportunity for a richer sharing of knowledge and understanding in a collaborative decision-making environment such as a command center.

**Table 1: Comparison of Information-Sharing Approaches**

Approach		Benefits	Limitations
<b>Pixels</b>		<ul style="list-style-type: none"> <li>Minimal requirements on receiving end: just a screen</li> <li>Straightforward to implement: video cables and switches</li> <li>Fewer security challenges</li> </ul>	<ul style="list-style-type: none"> <li>View cannot be reformatted for different target devices</li> <li>View cannot be modified</li> <li>View cannot be integrated with others</li> </ul>
<b>Data</b>	<pre>&lt;Threat&gt; &lt;name&gt; SAM-21 &lt;/name&gt; &lt;/Threat&gt; ...</pre>	<ul style="list-style-type: none"> <li>View can be integrated with others on the receiving end</li> <li>Potential bandwidth savings</li> <li>View can be reformatted for different target devices</li> <li>Can be used to enable role-based data filtering</li> </ul>	<ul style="list-style-type: none"> <li>Receiver must understand data format</li> <li>Potential challenges with transport across security boundaries</li> </ul>
<b>Recipes</b>	<pre>Select * from ThreatDB where Lat &gt; 40 and Lat &lt; 50 and Lon &lt; 45 ...</pre>	<ul style="list-style-type: none"> <li>View can be integrated with others on the receiving end</li> <li>Potential bandwidth savings</li> <li>View can be reformatted for different target devices</li> <li>Can be used to enable role-based data filtering</li> <li>Receiver can modify view</li> <li>Receiver can ensure relevance of results</li> </ul>	<ul style="list-style-type: none"> <li>Receiver must understand data format</li> <li>Potential challenges with transport across security boundaries</li> <li>Receiver must be able to reconstruct data view via access to original data sources</li> </ul>

This paper describes an effort to build a concept demonstrator that explores these higher levels of information-sharing – data and recipes – to support information integration and collaborative visualization in network-centric command centers.

## 2. Related Work

The HCI concepts developed for the prototype system build upon several past studies in the literature in the area of large screen interface design and interaction, multi-monitor displays, and the interplay between public and private displays. Here we provide an overview of the research that most directly influenced this prototyping effort.

The preponderance of human/computer interface design and evaluation studies in the literature focus on single-screen displays that are viewed at a typical reading distance. There exists considerably less guidance on how to make effective use of large screen displays viewed or interacted with at a distance, or multi-monitor display configurations on the desktop. However, some trends and insights have emerged in recent years. For example, studies conducted at Microsoft Research (Czerwinski, 2003) provide evidence from user studies on multiple monitor use indicating that conventional desktop graphical user interface (GUI) metaphors are not ideal for multi-monitor displays that provide a much larger-than-normal “desktop.” Their users struggled with the location of fixed screen elements such as the Windows taskbar and “Start” menu, and the size of the mouse pointer.

Somervell *et al* (2003) developed design heuristics for large screen displays by using scenario based design to examine five different types of large screen displays. Their heuristics include the following:

- Appropriate color schemes can be used for supporting information understanding
- Screen layout should reflect the information according to its intended use
- Judicious use of animation is necessary for effective design
- Large screen displays should show the existence and presence of relevant information, but not all of the details
- Configuration controls should not be placed on large screen displays, since they are largely used as a presentation mechanism

Guimbretiere *et al* (2001) and Rekimoto (1998) both emphasized the need for a ‘clean’ large screen display without widgets or any other current desktop GUI elements. Guimbretiere integrated interaction into the display itself through touch screen manipulation and floating menus. Rekimoto describes a system that removes all interactive elements from the large display. A pen device on a tablet computer is used to drag and drop elements directly between screens.

A related concern is how to allocate content and functionality across multi-display configurations, some of which may be private to one user and others shared and accessible by multiple users. Myers (2001) explored the use of a handheld computer to control a shared display. Such “multi-machine user interfaces” are defined as the distribution of computing functions and related user interfaces across all computing input/output devices available to the user. He suggests that the challenge in application design is to show only the appropriate information in each place and allow the fluid transfer of control and information between private and shared displays. He uses the scenario of a group of people collaborating during a lecture to exemplify how multiple handhelds can share and control the contents of a public screen.

Another challenge concerns how to enable interactions between displays: controlling system operation, moving information or content from a private display to a public display or vice versa, or moving information between displays at comparable levels of privacy. Various methods for

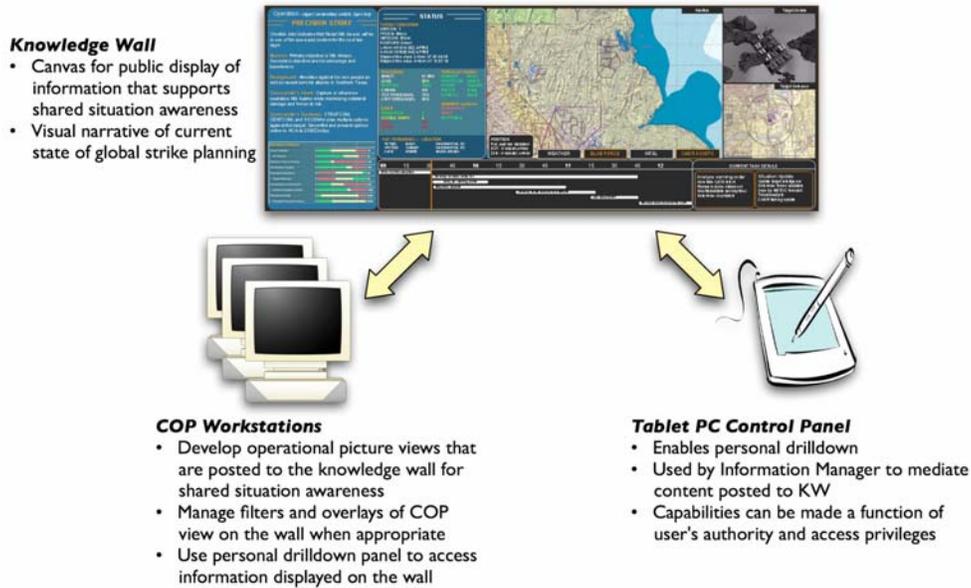
promoting information from personal displays to large screen public displays have been outlined (Swaminathan, 1997). One of these methods is called the *dollhouse* metaphor, which is an area on a personal display that represents a scale model of the large screen display and the applications or objects visible on it. Users select an application to interact with by choosing it on the dollhouse, which then launches a local application that communicates with its counterpart on the large screen display. Interactions within the dollhouse are then re-interpreted to represent interactions with the corresponding object on the large screen.

Many of the papers cited above describe single-point solutions that attempt to foster a closer partnership between large screen displays and other categories of computing devices such as handhelds or personal computers. Relatively little attention has been given to identifying and developing an overarching framework for multi-device interaction and information exchange. An exception is the work of Raghunath *et al* (2003), which creates a vision of *symbiosis* between handheld computers and other classes of display devices. Symbiosis is a term borrowed from the world of biology, used to describe a mutually beneficial relationship between dissimilar organisms. In this vision, some of the limitations of handheld computers – small screen space and limited interaction means – are overcome by developing effective partnerships between these devices and other technologies without these limitations. The current work expands on this concept by exploring how to share information between personal and public information systems in a military command center environment through such partnerships.

### 3. Human/Computer Interaction Concepts

#### 3.1 Members of the Ensemble

Figure 2 illustrates the members of the *SIDEView* (Symbiotic Display Ensemble for Visualization and Interaction) prototype. The centerpiece is the large screen **knowledge wall** (KW) display, which provides an electronic canvas for public display of information visible to everyone. The screen layouts developed for the KW provide a visual narrative of the state of a hypothetical strike mission planning scenario. The initial design provides a fused map display, a task timeline viewer, and an overview display of the current mission's objectives and evaluation criteria for developed course of action options.



**Figure 2: Members of the Prototype Ensemble**

The battlespace views displayed on the wall are developed on multiple **Common Operational Picture (COP) workstation** instances. Users develop battlespace views (e.g., red force intelligence, blue force readiness, weather, etc.) that are published to the wall using the recipe-sharing approach described earlier. Users can also drill down into information displayed on the wall via these workstations, using a customized interface for such interaction.

Finally, a **tablet PC control panel** provides an interface that an authorized *information manager* can use to manage the content displayed on the knowledge wall and to perform drill-down interaction with the wall's contents. Although there is no strict requirement that the control panel's functionality must be on a tablet PC or other portable computer, using it illustrates how a user that is mobile in the command center might interact with different information systems.

The knowledge wall is the central element in the display ensemble. Ideally, it should support operations by providing a canvas for displaying information that is of general relevance to the collaborative decision-making community in the command center. Several important design drivers for the large screen display can be identified through principles of graphic design, research in ergonomics and cognitive psychology, and an understanding of the command center environment. Figure 3 illustrates how these design elements come together in the overall layout of candidate knowledge wall designs.



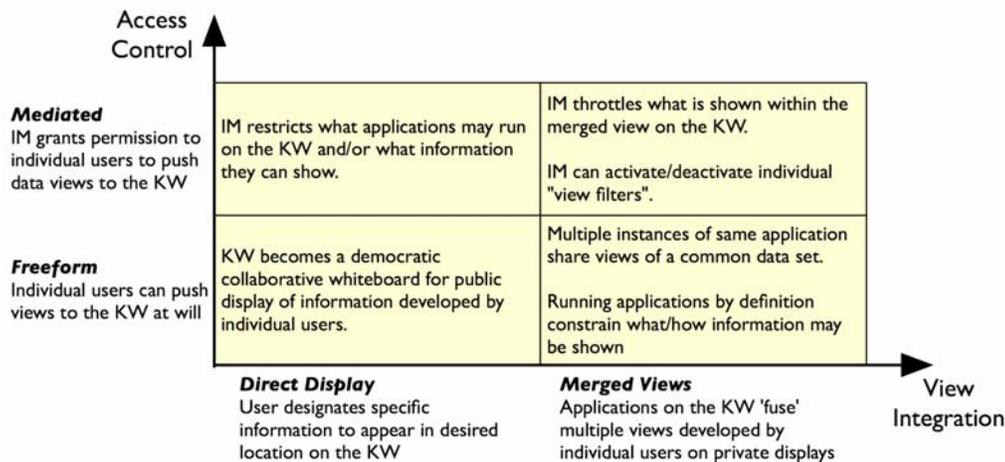
knowledge wall. Section 3.2.2 describes the concept of personal drilldown; i.e., pulling information displayed on public displays onto a private display for more detailed inspection.

### 3.2.1 Content Integration on Large Displays

The SIDEView concept calls for detailed mission views developed on individual workstation displays being “pushed” to public knowledge wall displays via the recipe sharing concept discussed earlier. Apart from the network-oriented transaction that takes place to accomplish this sharing (discussed in greater detail in the next section), there is also the question of how this sharing is managed from a human/computer interaction perspective.

We believe that there are two drivers to consider when determining how to manage the content that is pushed to a public display from elsewhere. The first is the degree of *view integration*; information from multiple different users can be fused into a common view on the large screen, or each information set can be displayed separately alongside the others. To reduce cognitive load in synthesizing multiple data sets, merged views may often be desirable. However, some categories of information may not be amenable to merging. In those cases, parallel display of each set may be appropriate. The second driver is the type of *access control*; information display can be controlled by an *information manager* (IM) who has the authority to regulate what is put on the knowledge wall, or it may be the case that no such mediation takes place.

Figure 4 illustrates how different combinations of view integration and access control gives rise to four distinct styles of interaction between personal displays and large screen displays. This not an exhaustive taxonomy; along each axis there could be incremental steps that go from freeform to mediated access control, and direct to merged views.



**Figure 4: Access Control and View Integration Matrix for Large Screen Interaction**

As information is moved from personal workstations to large screen displays, it becomes visible and accessible to a larger number of users. As such, the data displayed on these screens should be increasingly well-organized to facilitate rapid comprehension. Data must be laid out in an increasingly structured style and interaction between viewers and the display is likely to become less direct. These requirements can be achieved by mediating how information appears on the large screen - introducing *information managers* who can authorize data for display and perhaps control how it is integrated onto the large screen display. The information manger can mediate the display of content to make sure it is appropriately verified and suitable for public display.

Absent such mediation, a **freeform direct display** interaction allows any user to put information on the large screen. Each item will appear as a separate entity, perhaps in its own application window. If this interaction style were implemented on a large screen, its display would become a democratic collaborative whiteboard for public display of information developed by individual users. This interaction style might be useful in a small group situation, where information has not been fully vetted but is still in the process of collaborative development.

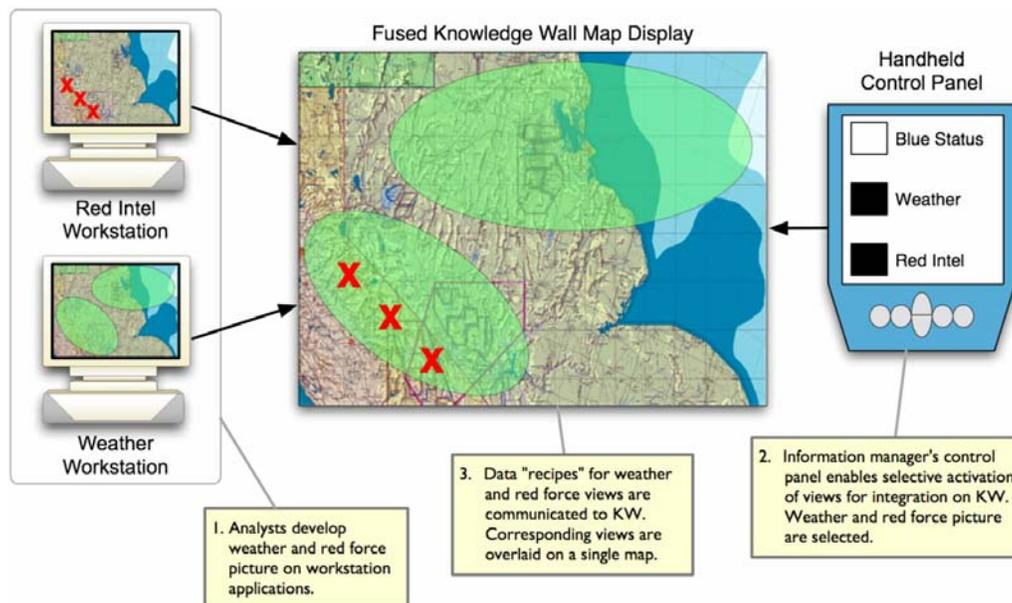
A **freeform merged view** display interaction will still allow any user to put information on the large screen, but the large screen views will be designed to fuse multiple individual views into a single space. This interaction style will be useful when multiple instances of the same application running on separate personal computers can combine their individual parts into a shared whole. For example, in a mapping application it may be desirable to combine the views developed by multiple individuals into an integrated set of overlays atop a single map on the large display.

The top left corner of the figure describes **mediated direct display** interaction. This approach restricts the permission to display information on the large screen to a designated information manager, who will determine what should be displayed, and then designate its location or grant permission to the originator of the information to place it on the large screen.

Finally, a **mediated merged view** enables the information manager to fuse the information into a single presentation using a control panel on their personal computer. This interaction style will be useful when multiple instances of the same application running on separate personal computers can combine their data parts into a shared whole. For example, in a mapping application it may be desirable to combine the views developed by multiple individuals into an integrated set of overlays atop a single map on the large display.

Each of the four interaction styles has their appropriate uses in a symbiotic display environment, driven by the type of information being displayed, the capabilities of the applications running on the various displays, the type of activity that is the focus of attention, and the CDM environment's culture.

As an example, Figure 5 illustrates a potential use of the mediated, merged view: integration of individual map views into a common map display on the large screen. The workstation-level views developed by individual analysts are integrated onto a merged knowledge wall map display under the direction of the information manager. This visual fusion allows decision-makers to see the potential interaction between enemy activity and weather conditions, without having to perform a mental fusion of the contents of the individual workstation displays.



**Figure 5: View Integration Concept**

Within the software, information is pushed from workstation displays to the wall display at the *recipe* level: rather than shipping pixels or even the data representation of weather/intel information, the workstation tools push the recipe for constructing these views to the wall display. These recipes are the database queries and filters used to construct the workstation views. Upon receiving them, the wall display reconstructs the same picture by accessing those same information sources.

### 3.2.2 Drilldown from Public Displays onto Personal Displays

Once information has been integrated onto the knowledge wall, there is a question of how to interact with it and inspect its underlying details. On a workstation display one could simply pull up drilldown windows that illuminate the underlying details on information of interest, but this is somewhat more complicated when viewing a knowledge wall at a distance. The mouse and keyboard associated with the wall display may only be accessible to one command center operator. That operator could pull up drilldown displays and show them on the wall directly in response to a verbal request from a senior decision-maker, but this has two limitations: (a) the drilldown windows may interfere with other people's views of whatever is behind them; and (b) the detailed information may be difficult to digest at a long distance.

We believe that these limitations can be overcome by the concept of "personal drilldown:" when a user sees something on a large display that they would like to explore in more detail, they can pull that information down to their workstation and view it there. They use an area on their personal screen that represents a scale model of the large screen display and the applications or objects visible on it (called a *dollhouse*) (Swaminathan & Sato, 1997) to identify and retrieve that data. Users select an application to interact with by selecting it on the dollhouse, which results in a communication with its counterpart on the large screen display. Through that local application, the user has access to the detailed information associated with the summary shown on the large screen. The information the user receives can be role-based; they may receive information tailored to their interests and privileges. They can then browse the underlying details of that information on their local display.

Figure 6 illustrates this personal drilldown process on the tablet control panel. The lower part of the left figure shows a dollhouse representation of content shown on the wall display. In this example, each individual application on the wall has a corresponding button on the tablet control panel. Tapping any of the buttons initiates a network transaction used to retrieve a URL for direct interaction with the application on the wall (the implementation details are discussed in the next section). The contents at that URL are then displayed within a web browser on the tablet PC. This browser then reproduces the information shown on the knowledge wall on the user's local display, for more detailed inspection and browsing of underlying mission details. The user can click on any of the hyperlinked information shown and browse the underlying page on a local web browser display, as shown.

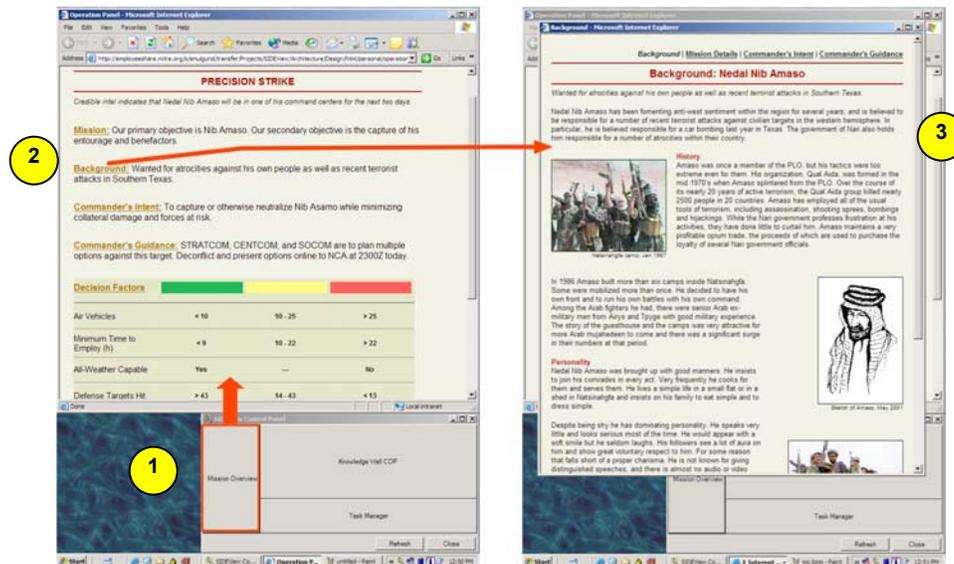


Figure 6: Personal Drilldown using Dollhouse Control Panel

## 4. System Architecture

### 4.1 Overview

In developing the knowledge wall applications that would support symbiotic, information-sharing partnerships with applications on other displays, there was some question as to how to expose those capabilities, and then allow clients to find them. As noted by Raghunath *et al* (2003), options for discovery and association include universal plug-and-play, Sun's Jini technology, and the universal description, discovery, and integration (UDDI) protocol for web services. Non-networked options could include direct communication between systems using infrared transceivers. A network-oriented solution was chosen for this prototype, to support high-bandwidth data transfer when needed (as well as non-line-of-sight interaction).

The implementation options for providing remote interactions include web services or distributed object technologies such as Jini, Java Remote Method Invocation (RMI), Microsoft's *.NET Remoting* (Rammer, 2002), or the Common Object Request Broker Architecture (CORBA). Web services were initially appealing because they use a standardized representation for data exchange – the extensible markup language (XML). However, web services are traditionally hosted in web servers, not standalone applications with a graphical user interface. Although there are now emerging tools that permit the deployment of web services independently of any web server (Vogels, 2003), distributed object technologies were determined to be a better fit for this

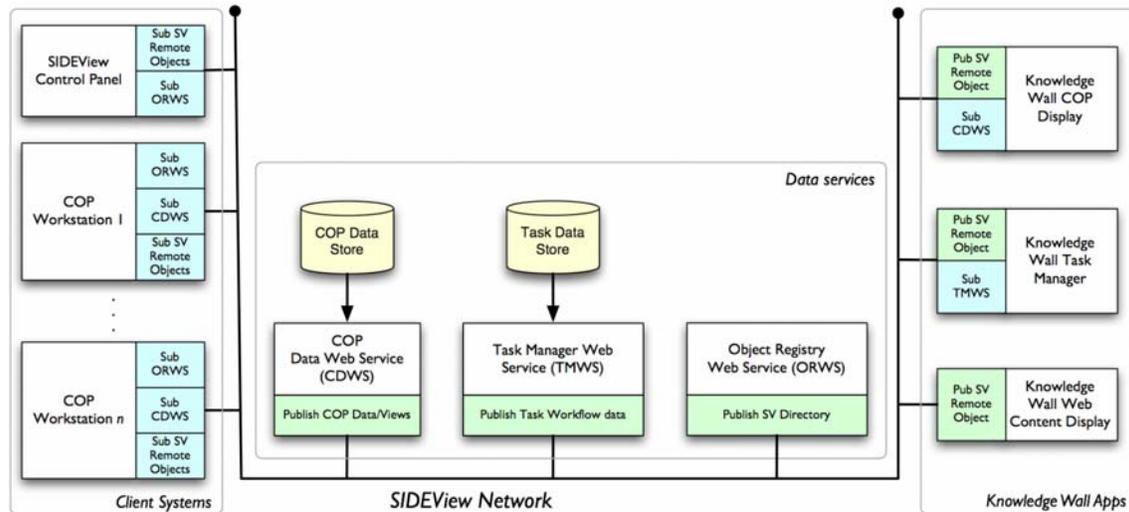
effort. Microsoft's .NET remoting framework was subsequently chosen, because of the ease and speed with which it could be used to create and test concepts for inter-application communication and messaging.

Based on this decision it was necessary to identify a mechanism for client devices in the environment to find and form associations with the pertinent KW applications. Technologies such as UDDI apply to discovery of web services; they were not designed for remote objects *per se*. The .NET remoting framework provides built-in mechanisms for clients to find remote objects on the same physical machine, but this doesn't readily extend to situations where clients may not know the network address of the resource with which a partnership is desired.

A simple solution was adopted here to facilitate rapid prototyping: an XML web service was built to act as a registry of remote objects providing information-sharing services. Applications providing such capabilities register themselves with this web service upon instantiation, providing a URL and an interface specification that clients can use to communicate with them. Client applications on personal displays then seeking to communicate with the KW applications only need to know where to find this web service, from which they can locate and connect with those applications.

Figure 7 illustrates the prototype's logical software architecture. As shown, there are three key components: knowledge wall applications that publish symbiosis-enabling remote objects, client systems (running on personal computers, tablet PCs, etc.) that may wish to interact with those services, and finally, XML web services needed by all applications. The Common Operational Picture (COP) data web service (CDWS) provides simulated data on friendly forces, threats, weather, etc. It also provides a repository where the users of individual COP workstations can publish the filters and queries they have developed to interpret that data. The objects registry web service (ORWS) provides the mechanism by which client applications find the symbiotically enabled applications on the knowledge wall, as discussed above. This is part of the discovery and association process needed to enable inter-application information-sharing. Finally, the Task Manager web service (TMWS) provides a simulated repository of current workflow activities.

Each of the three applications on the KW – the integrated map display, the task timeline viewer, and the mission overview viewer – publish remote objects that are exposed through .NET remoting. These remote objects register themselves with the objects directory so that clients can find them. Finally, on the left are the client systems that may wish establish symbiotic relationships with the KW applications. Each of these systems subscribes to the web services for COP data and the remote objects registry. They also connect with the remote objects published by each of the KW applications.



**Figure 7: Prototype Architecture**

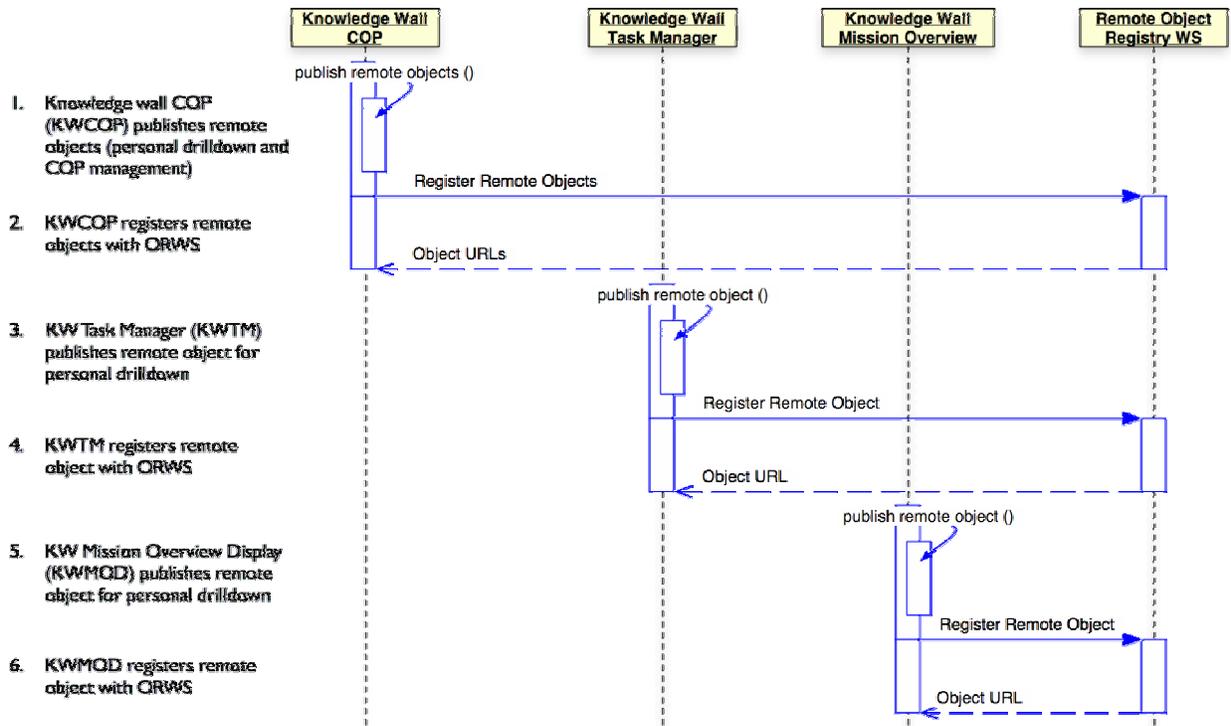
The specific remote interaction capabilities provided by each of the KW applications are a function of the particular application. For example, the KW COP display provides for mediated merged views. This requires exposing mechanisms for clients to request that their view be incorporated into the public map, as well as providing the means by which the information manager can grant/deny those requests through the personal control panel. The mission overview on the KW is a web browser window stripped of all GUI widgets. It supports personal drill-down, so that command center occupants can view the same information on their personal display, along with supporting details. In this case, the inter-application messaging allows clients to ascertain the web URL used to display the content. Note that when users view that information locally, it may be reformatted for the constraints of desktop displays.

## 4.2 Inter-Application Data Flows

The individual SIDEView applications, web services, and remote objects all work together to enable the view integration and personal drilldown capabilities described earlier in section 3.2. Here we describe the inter-application instruction and data flows that enable this behavior. As will be seen, the “push” and “pull” capabilities are enabled by a loosely coupled set of interactions between individual applications, distributed objects, and web services.

### 4.2.1 Remote Object Registration

Each of the SIDEView knowledge wall applications that provides remote interaction capabilities must declare the existence of those capabilities with the ORWS described above. Figure 8 provides a Unified Markup Language (UML) sequence diagram of the interactions that take place during this registration process. The particular order in which the applications register with the ORWS is not important; that depends entirely on the order in which they are launched on their host platform.



**Figure 8: Sequence Diagram of Remote Object Registration**

As shown, the KWCOP publishes two remote objects: one for personal drilldown and another for enabling the COP view/filter management capability. It then contacts the ORWS and provides the salient details that describe each of these objects. The ORWS returns a URL that conforms to .NET remoting protocols for communicating with the remote object, which will be used by clients wishing to interact with it. This process takes place for each of the objects published by the KWCOP. The KW TSM and KW MOD then perform the same publish and registration process for their personal drilldown remote objects. Once this process is complete, the remote objects are live and ready for interaction with remote clients. The next step is for those remote clients to discover and invoke them, as discussed below.

#### 4.2.2 Remote Object Discovery

Once the KW applications have published and registered their remote objects with the ORWS, remote clients can locate and interact with them. This is the process used to construct the “dollhouse map” of the knowledge wall, which was illustrated earlier in Figure 6. This makes the personal drilldown interaction shown in that figure possible. Figure 9 presents the UML sequence diagram of the process by which the personal control panel configures this dollhouse representation.

