Location Independent Mission Operations – A Systems Engineering Approach to Mobile Device Data Dissemination

Edward Birrane[1] and Robert Berardino[2]
Johns Hopkins University Applied Physics Laboratory, Laurel, MD 20723

Mission operators must command, diagnose, and correct in-flight spacecraft in a timely and cost-effective manner. These functions may involve near-real-time coordination with flight hardware/software engineers, test teams, instrument subcontractors, scientists, and other subject matter experts not typically present in the Mission Operations Center (MOC). During regular and emergency operations, missions benefit from the ability to connect geographically disparate resources with near-real-time (NRT) mission data. Highly mobile, computationally capable tablets, such as iOS and Android devices, enable this novel data mobility in an affordable and scalable way. However, this flexibility is not achieved by simply treating these devices as standard Internet devices. Rather, they require an evaluation of data and control flows through the MOC architecture. Security models must implement access controls to protect data access from outside of MOC subnets and limit exposure of flight networks to public networks. The touchscreens that comprise the main user input of tablet devices require a human-computer-interface analysis to visualize verbose, multi-dimensional telemetry. Finally, local computation and data storage must be optimized to provide reasonable battery life for these energy-constrained devices. This paper outlines the benefits of mobile devices in MOCs, including a series of operational use cases uniquely enabled by this technology. We present a set of metrics, such as Mean Time To Data Expert (MTDE), used to measure the benefits to adopting missions. A multi-tiered data distribution architecture, as prototyped at the Johns Hopkins University Applied Physics Laboratory, is presented along with lessons learned from the implementation. We conclude that mobile devices will continue to grow in capability, lessen in cost, and will inevitably migrate into the mission operational picture. However, without the systems analysis and design activities outlined in this paper, initial implementations will risk security, usability, and stability issues for early adopters.

Nomenclature

API	=	Application Programming Interface
COTS	=	Commercial-Off-The-Shelf
DMTE	=	Data Mean Time To Expert
HCI	=	Human-Computer Interface
LIMO	=	Location-Independent Mission Operations
MOC	=	Missions Operation Center
NRT	=	Near Real Time
PMCD	=	Personal Mobile Computing Device
REST	=	Representational State Transfer
T&C	=	Telemetry and Command
UX	=	User Experience
VPN	=	Virtual Private Network

[1] Senior Staff, JHU/APL, Space Department, 11100 Johns Hopkins Rd, Guest Member
[2] Senior Staff, JHU/APL Space Department, 11100 Johns Hopkins Rd, Guest Member

I. Introduction

The role of mission operator involves the command, diagnosis, and correction of in-flight space assets. Based upon mission characteristics and severity of fault, operators must accomplish these tasks with varying degrees of timeliness and often within constrained budgets. When responding to mission events, operators may require coordination with subject matter experts not normally present in a Mission Operations Center (MOC), including flight hardware/software engineers, test teams, instrument subcontractors, and scientists. It is a common occurrence to require a variety of support staff to be physically present in MOCs during critical mission events and maneuvers; there is also precedent for engineering support to be called to the MOC at any time, day or night, to assist in responding to unexpected events. Critical mission events, expected or otherwise, will always require infusions of expertise. However, advances in mobile computing, broadband infrastructure, and software frameworks may reduce the logistical cost associated with this support. Specifically, removing the requirement for support activities to only occur in the MOC reduces cost (experts do not need to be idle in the MOC when not needed), reduces logistical challenges associated with synchronizing availability, and reduces delays in getting expert dispositions on unexpected events. We term this set of capabilities Location-Independent Mission Operations (LIMO).

The challenge associated with implementing LIMO capabilities stems from the evolution of supporting technical means. Simply adding laptops and mobile devices behind Virtual Private Network (VPN) connections in support of legacy data and control flows will not effectively solve the location-independent issue. A systems engineering approach is necessary to both identify new data and control flows *and* demonstrate how these flows provide benefits to adopting missions.

We propose that the continued evolution of tablet computers and broadband access provides a new technical means making feasible the command and control of space assets with appropriate data rates and security models. We further propose that these evolving technical means require changes to existing operational concepts and infrastructure to be used effectively. Efforts to incorporate mobile devices must account for their unique characteristics in the areas of input methodology, screen size, battery life, and security models. Server-side infrastructure must be built to perform data conditioning, user registration, preferences processing, and validation. User applications must undergo human-computer interface (HCI) analysis to determine how to maintain a consistent "look-and-feel" across a variety of platforms to reduce training and support costs. Ultimately, this analysis may change the number and nature of applications supporting a mission. We predict that the flexibility and reduced operation and maintenance cost associated with a distributed data model will recoup initial systems engineering and architecture investments.

The remainder of this paper is organized as follows. Section II details the motivation of our approach, both in the use of new technical means and in the need to re-evaluate flows in the system. Section III presents an initial systems engineering analysis of a LIMO capability, including the characteristics, operational concepts, and metrics used to evaluate technical means. Section IV provides a candidate architecture conformant to our system-level concepts. Section V discusses our experiences with a reference implementation of LIMO capabilities for operational missions. We summarize our work in Section VI.

II. Motivation

In this section, we consider smart-devices in the broader context of computing machinery to understand their unique role in bringing computational benefits to operational systems.

The laptop computer represents the latest achievement in a tradition of computer miniaturization that has, prior, seen the migration of user computation from mainframes to servers to desktops. Each new technical achievement in this area makes the same HCI models of the mid-1980s[1,2] available in different cost/mobility profiles. To date, reducing the cost and administrative complexity of these systems has increased re-use and decreased the cost of ground systems. Standardized applications for complex visualization, data sharing, automation, and communication reduce operator error and increased the ability of operators and systems engineers to respond to problems more quickly and with more accuracy. Further, the re-use profile for ground systems in this architecture is very high; applications are not hard-coded to a particular server, desktop, or laptop computer. However, this legacy HCI hinders the continued miniaturization of the general purpose computer; keyboards must be of a minimum size for typing, screens must be of sufficient size to accommodate standard windowing views; mice/trackpads must be of usable size.

Hardware capabilities continue to mature, enabling increased battery density, low-power-consumption components, touch/gesture screens, and overall reductions in cost. Hardware advancements have created a new category of computing device which, by necessity, deviates from the legacy HCI model. These devices are referred to as "tablets" or "smart phones" and contain gigahertz CPUs, gigabytes of RAM, individual graphics processors

and gigabytes of persistent storage. The continued maturation of wireless technologies connects these devices at broadband link speeds. Collectively, we refer to them as "Personal Mobile Computing Devices" (PMCDs). PMCDs offer a significant, new user capability: personalized, mobile data consumption. Because they introduce changes to the HCI, their incorporation into existing systems requires analysis.

Typical sizes of PMCDs preclude built-in keyboards, mice, or multi-windowed screen displays. Touchscreen inputs impose minimum sizes to interactive widgets based on average finger sizes[3]. Gesture controls impose a physical manipulation metaphor that changes the familiar button-click view model[4]. Beyond changes to the interface, PMCDs function differently from desktop and laptop computers. Their smaller size limits effective battery life and secondary storage. Their slim designs and lack of active cooling present thermal considerations when performing sustained computation. Therefore, while PMCDs support powerful processors, the loading of these processors is dedicated to low-power operation, monitoring of thermal environment, and a priority towards system-level scheduling and touchscreen servicing versus user application number crunching. The relative strengths and weaknesses of currently offered PMCDs are listed in Table 1.

Table 1. Applications must account for both strength and weaknesses of PMCDs

Functional Area	PMCD Strengths	PMCD Weaknesses
Processing	Multi-core, Ghz processors and Gb of RAM	Embedded processors with reduced instruction sets; high cost of polling/decoding complex touchscreen gestures; thermal issues via lack of active cooling
Power	Dense, small batteries last for days compared to hours for laptop devices.	Battery life drops dramatically under moderate, persistent computational load.
Storage	Very fast, solid state internal storage typically supporting additional Secure-Digital (SD) cards.	Less than 80Gb total storage on most devices, even using largest SD cards. Not all devices support external storage media.
Networking	Wireless/3G/4G network availability.	Typically no wired network access.
Input Devices	Touchscreen, some support for Bluetooth keyboards and mice.	Not all devices support keyboard input. Keyboards and mice not typically available for use when mobile.
Display Sizes	High-resolution, lightweight, rugged displays.	Very small size (3-10")

As ground system architects seek to reduce cost and increase capability there is a natural desire to evaluate new technologies. Since a significant portion of ground operations relies on communication with appropriate resources at appropriate times, the unique benefit of increased mobility has increased interest in PMCDs for operational use. However, beyond the desire to incorporate socially popular devices into operations centers, we have seen no systematic analysis of the unique functional benefits of these devices, what architectural changes must be made to MOCs to realize those benefits, and whether the benefits justify the associated cost and risk. To date, many commercial vendor ground systems incorporate PMCDs as if they were any other remote thin-client, with few architectural changes to support their unique properties. There is acknowledgement that alternate graphical user interfaces are necessary, but the quality and consistency of these interfaces varies widely.

We term this the "bolt-on" approach and while it *is* a low-cost way of incorporating socially popular devices into operations centers, it *is not* an efficient way of achieving the operational goal of leveraging data mobility. As PMCDs grow in popularity, increase in computation capability, and decrease in cost, their role in mission operations will increase. However, architectures must adapt to the relative strengths and weaknesses of this emerging technology means. The inappropriate focus on the popularity of PMCDs, rather than analysis of how they evolve operational concepts, motivates our work. We seek to describe the unique characteristics of this new category of computing device and perform a systems-level assessment of how to appropriately incorporate it into ground-system architectures.

III. Systems Analysis

This section summarizes our systems analysis associated with the characteristics of PMCDs in MOCs, how such characteristics benefit mission operations, operation concepts to achieve these benefits, and what metrics justify the expense of architecting PMCDs into operational systems.

A. Characteristics

From Table 1 we decompose the concept of "mobility" into three unique characteristics governing the development of applications for a PMCD: screen size, alternate input methods, and processor/storage. We add the fourth characteristic of security to the list as a special consideration for passing operational data outside of a dedicated flight network. While we discuss security only from the point of view of MOC networks, we recognize there is a very large body of research regarding the securing of data and services for PMCDs for any application[5,6]. The manner in which each of these characteristics informs the application architecture is as listed below.

1. Screen Size

In support of lower power consumption and higher mobility, screen sizes for PMCDs are significantly smaller than those for wall monitors, desktop displays, and laptops. Therefore they cannot support the same level of information density in their displays. This constraint requires developers to reduce the "footprint" of information on the device, either by using pictures/graphics in lieu of words, or by displaying smaller data sets. In either case, the associated changes to the user experience (UX) levies requirements on the architecture and protocols used to generate data for the device, especially when the device will contain fused or otherwise summarized information.

2. Alternate Input

PMCDs struggle to support general-purpose inputs. Keyboards and mice are typically too bulky to physically couple with the device, mechanical parts wear excessively on mobile devices, and relying on wireless peripherals requires carrying additional equipment and dedicating physical workspaces to their use. These issues are counter to a hand-held mobility model. On-screen keyboards take up valuable screen space and result in slow, error-prone inputs. PMCD developers exploit gesture-enabled touchscreen devices to provide interactive controls to manipulate data and control in their applications.

A reliance on constrained, interactive controls over free-form text input aligns precisely with the existing practice of building syntax checkers, command compilers, validators, and other tools to prevent user error in the MOC. In this regard, the evolution of error-correcting input methods provides a compelling alternative to the more complex task of verifying free-formed user data.

3. Processor/Storage

Hardware resources in PMCDs are not used in the same way as on traditional computers; storage is orders of magnitude smaller and processing is constrained by thermal considerations, battery density, and secondary loads from the touch-screen. Applications running on PMCDs lack the working memory, battery life, and persistent storage to perform sustained, complex computation. Where moderate computation occurs for extended periods of time, even state-of-the-art devices experience thermal issues.

Applications built for MOCs vary from simple data conditioning tasks to complex visualization and calculation tasks. While some applications may be ported directly to a PMCD after changes to the user interface, others must be completely re-architected around these limitations. For example, many mobile application developers support a thin-client architecture where server-side infrastructure performs complex computation and sends processed results to the mobile device for visualization. Understanding how MOC utilities and visualizations map to this or other models is a key activity when incorporating PMCDs into the ground segment architecture.

4. Security

The literature is densely populated with wireless security models operating at every level of the networking stack. Implementations of secure ciphersuites can be ported to PMCDs, especially on institutionally administered devices. However, the implementation of security protocols, and the security of data at rest, remains inconsistently implemented across various vendors. Part of this stems from constraints in the device hardware and part from the level of security in the implementing operating system and application software.

Adopting PMCDs for use in flight environments requires that ground segment architectures adopt policies to standardize the security model across all supported devices, including implementation of security at the individual application layer if not otherwise provided by a particular operating system/hardware combination.

B. Benefits

Based on these characteristics, and experience building applications for MOCs, several potential benefits arise from the incorporation of PMCDs into the ground architecture. Of interest, none of these benefits are unique to PMCDs, but their mobility, intuitive interfaces, and focus on personal data consumption provide novel ways to

achieve long-standing application goals. This section describes three benefits identified through our analysis, as identified in Table 2.

Table 2. PMCDs provide three key benefits to mission operations support.

Benefit	Associated Characteristic			
	Screen	Input	CPU/ Storage	Security
Decreased operator error through natural input methods and focused visualization.	●	●		
Faster anomaly resolution with coordination amongst geographically distributed experts.				●
Reduced training cost via consistent interfaces and configurations across set of interfacing devices.		●	●	●

1. Decreased Operator Error

Traditional MOC displays are dominated by large screens that provide multiple dimensions of information, certainly more than an individual operator can digest in real time. This is a time-saving convenience when screens are shared/projected for view by multiple operators at once. Similarly, tools for command construction benefit from the availability of nearby, multiple computer systems to visualize flight rule/constraint documentation, perform simulation runs, and validate command sequences. Conversely, the smaller screen size of the PMCD is meant for individual viewing of information, and often in a more graphical form.

Therefore, smaller data sets must be constructed using filtering, fusion, and other aggregation methods to present targeted information in a smaller footprint. This reduces the cognitive load of the operator by focusing only on those data associated with a particular function at a particular time. Further, the growing practice of providing graphical widgets, pre-selected options, and other means to select amongst a finite set of options removes free-form text-entry as an input choice. When operators can focus on simplified views of their data and interact with it in natural ways by selecting amongst pre-configured options then opportunities for error decrease.

2. Faster Anomaly resolution

Robust security models, combined with increasing wireless speeds, enable NRT data interaction with devices outside of the MOC enclave. Authentication ensures that data provided, and controls received, are in accordance with access control policies configured on the system. Integrity ensures exchanged data is unmodified/uncorrupted. Confidentiality prevents unauthorized users from accessing these data while in transit. The ability to provide these levels of security allows multiple, geographically separated experts to coordinate in support of mission operations events with NRT access to telemetry and other spacecraft state.

Anomaly resolution activities, therefore, are no longer time-constrained by delays in getting experts into the MOC, delays in releasing information to remote experts, and delays in applying expert-recommended configurations and controls. This ultimately allows the anomaly resolution process to begin sooner, and with more focused input.

3. Reduced Training

The use of thin-client architectures on PMCDs, coupled with the enforcement of roles and responsibilities from a security layer, provides a consistent visualization and processing context to users. This experience can be migrated to any PMCD, laptop, or desktop supporting the thin-client application. By storing user-preferences and configuration information on server-side resources, thin-clients across multiple devices do not need to be reconfigured. Preserving look and feel reduces the number of systems on which an operator must be trained. Enforcement of roles and responsibilities at the application layer further reduces training to just those areas of the system where a user has permissions.

Since PMCDs replace unnatural motions (point-click, double-click, right-click, drag-drop) with natural motions (swiping, pinching, multi-point scrolling) the learning curve for applications is further reduced.

C. Operational Concepts

We translate the benefits of PMCDs, derived from their characteristics, into a set of three discrete operational concepts: location-independent mission operation, HCI, and data filtering. These concepts capture the benefits of PMCDs as a series of tangible activities based upon our experience testing PMCDs in mission operations contexts at the Johns Hopkins University Applied Physics Laboratory (JHU/APL).

To provide context for these CONOPS, we contrive an operational scenario where a faulty heater is causing thermal imbalances on a deep-space asset. Mission operations are monitoring the heater and its affect on spacecraft autonomy in preparation for an instrument observation.

1. Location Independent Mission Operation

A mission operations PMCD begins the day as a wireless node on a personal home network (802.11n) communicating over the Public Internet. The operator checks heater state for the prior eight hours and notes some period of anomalous current draw through the night. The PMCD transitions to a cellular network (3G, 4G, or 4G LTE) during the operator's commute to work. Upon arriving to work, the PMCD transitions off of the carrier's network and onto either an institutional demilitarized zone (DMZ) wireless network or other institutional LAN subnet. During morning meetings, feedback on the PMCD alerts the operator to another heater anomaly. Curious to observe the event, the operator excuses herself and migrates to the MOC to discuss the event with other operators. Coming out of the MOC, the operator locates the hardware engineer associated with the heater and they spend time in her office reviewing telemetry and associated spacecraft state on the PMCD. The hardware engineer has the heater manufacturer log into a web-version of the thin-client from their offices in a different part of the country, and a teleconference occurs to discuss the *NRT* observations of the heater in-situ.

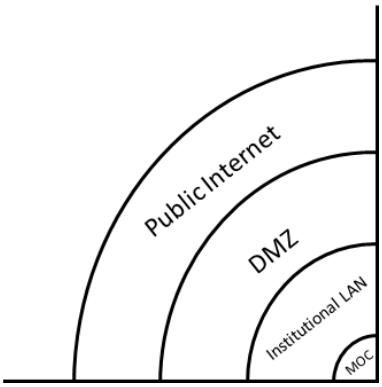

Figure 1. LIMO services operate across a variety of sub-networks.

Regardless of where the user physically travels throughout the day, a LIMO solution provides functionally equivalent application access regardless of the traversal through various sub-networks, illustrated in Fig. 1. Tracking radio hand-offs, authentication, and policies for data-at-rest are technical implementation hurdles. In this operational concept, public radio hand-offs are automatic and institutional access is granted by prompting the user to authenticate to the institutional sub-network on first data access. Authentication keys are maintained on the device itself as part of configuring the device. A default, and conservative, security policy requires that MOC data either not be serialized to flash memory, or be strong-encrypted prior to persistence. In either case, institutionally-approved thin-client applications conform to this policy.

2. Human-Computer Interface

Two popular operating systems for PMCDs, Google's Android and Apple's iOS[7], provide software development kits (SDK) enabling developers to decode touchscreen inputs including gestures. Further, best practices regarding the use of these input devices are available to maximize the use of these SDKs[8]. When designing a thin-client application for a PMCD, a software engineer utilizes these SDKs and best practices to develop graphical metaphors associated with the data manipulation.

For example, the data visualization of the heater telemetry in our scenario is a series of line plots over time. The operator interacts with these data in a variety of ways uniquely enabled by the touchscreen interface. Users may use single-touch selection of a data point, pinch-based zooming, and two-finger horizontal and vertical scrolling to adjust axis scaling. Data sets may be selected for inclusion in the graphing area by swiping. The type of data (raw, decommutated, averaged, weighted averaging, differencing) is selected by touching the data source icon for 1 second and then selecting type from a pop-up sub-menu of icons. A conceptualized view of this interface is shown in Fig. 2.

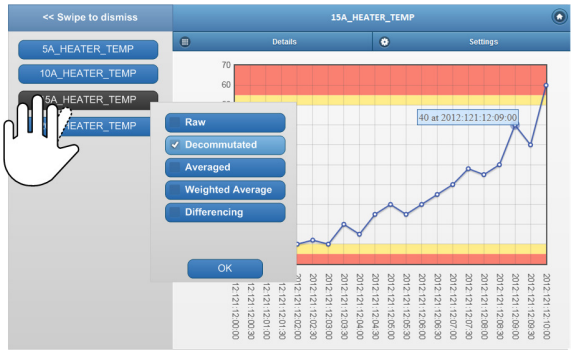

Figure 2. PMCD interfaces use natural gestures to choose amongst pre-selected data.

The user performs all critical visualization functions without the use of text-based input. The selection of data sets is as provided from a pre-defined list based on the user's role and functional areas providing the data. The application prevents incompatible data sets from display on the same graph and constrains the number of data sets that can be concurrently visualized. The process of data selection and visualization is not error prone, occurs rapidly and naturally with a minimum of training, and works the same way across a variety of data set visualizations. In fact,

the operator is able to perform visualization on the PMCD faster through the gesture interface than they would otherwise be able to using a mouse and keyboard interface at a laptop.

3. Data offload/filter.

The application software engineer allows users of the data visualization application to personalize their display settings by supporting customized navigation screens (e.g., a "My Favorites" page), remembering recent searches for data points to plot, default plot characteristics such as colors and units, and pre-configured plots of commonly visualized data sets. These settings are saved on the local device, and optionally on a user-preference setting at the application server.

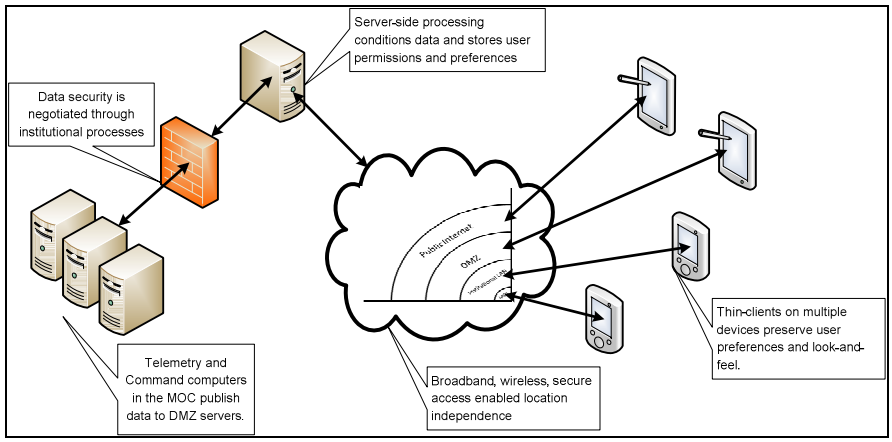

Figure 3. A multi-tiered architecture reduces reliance on a single PMDC.

In our scenario, as the operator continues to monitor the state of the heater and collects evidence for anomaly resolution, the PMDC is exchanged for another one with a fresh battery from a surplus of such devices kept in the MOC. For example, an iPad may be exchanged for an Android tablet. The operator takes two, providing one to the hardware engineer and takes the other one home at the end of the day. Neither of the two "new" PMDCs require configuration as they receive their configuration from the PMDC server; the views and pre-configurations of the heater telemetry data sets were saved on the original PMDC, migrated to the server side, and downlinked to the new PMDCs without significant user effort. An example architecture supporting these CONOPS is given in Fig. 3.

As the hardware engineer continues to evaluate past telemetry values, more complex data conditioning activities are requested, including three-day moving averages of the 1Hz data, histograms, and number of threshold crossings. Computing these across larger data sets on the mobile device is impractical from a time and battery-life perspective, so they are off-loaded to the server, as part of the application software, and results are displayed to the user when the computation is complete.

D. Metrics

In the set of operational concepts outlined above, there are three values that we can measure to quantify the benefit of the PMCD approach to mission operations: (1) the time it takes to identify data related to a problem, (2) the amount of time it takes to request and receive relevant data, and (3) the amount of time before subject matter experts can review data. These metrics are captured in Table 3.

Table 3. The benefit of PMDCs in the operational environment can be measured with three metrics.

Metric	Description	Value in Optimizing
Problem Identification Time (PIT)	The lag associated with collecting information from the spacecraft and identifying anomalies in that information set. Critical faults and large anomalies are captured automatically through threshold crossings. Subtle anomalies, such as those captured from trending, may require manual detection.	Subtle anomalies caught early may resolve with preventative, rather than corrective action. More rapid anomaly identification may reduce the change of failure cascades in flight systems.
Near-Real-Time (NRT) Deviance	The age of the data measured as the lag between when the data was received in the MOC and when it was available for consumption by experts.	Up-to-date data is required for health monitoring and anomaly resolution.

Data Mean Time to Expert (DTME)	The time between problem identification and evaluation of data by a subject matter expert.	Shorter DTME implies faster initiation of the anomaly resolution process.

IV. Data Delivery Architecture

A. Architectural Goals

Based on our systems analysis of the benefit of PMCDs to mission operations, we have constructed a reference data-delivery architecture and implementation at the JHU/APL. The goals of this reference architecture are provided in Table 4.

Table 4. Implementation and Support Constraints Drive Architectural Goals.

Goal	Rationale
Highly-Performing Data Distribution	LIMO data flows should achieve near real-time data processing. The round trip time includes a MOC to DMZ server data caching layer and DMZ server to PMCD time. When not communicating real-time data from a pass, the delay is effectively from the DMZ server to the PMCD.
COTS Vendor Agnostic	As COTS products, Telemetry and Command (T&C) products, and other services may vary amongst supported missions and implementing institutions, a vendor agnostic approach is required.
Wide PMCD Platform Penetration	Because of the multitude of existing devices, the proliferation of new devices and ever dynamic nature of the PMCD marketplace, the architecture must provide a way to make NRT data available to as many PMCD platforms as possible.

B. Related Products and Standards

The Object Management Group (OMG) published a Data Distribution Specification for Real-Time Systems (DDS) in 2004 to define an interoperability protocol for DDS. Its purpose and scope was to ensure that applications based on different vendors' implementations of DDS could interoperate. Commercial vendors like Real-Time Innovations (RTI) and PrismTech have DDS products that are implemented in accordance with the OMG DDS specification. This specification works to address the need for a common networking middleware and publish / subscribe model for data access in large distributed systems. The types of large-scale systems that utilize a DDS implementation include: Air Traffic Control, Railway Traffic Management, Radar Processors, Naval Combat Management Systems, and UAV Control. These systems could also benefit from extending data in a location independent manner.

Commercial T&C vendors offer products that extend their primary product in an attempt to extend an operation center's reach across network boundaries. Harris Corporation, L-3 Telemetry West, and Integral Systems, Inc. all have T&C products that support secondary "bolt-on" offerings to extend their T&C product into some form of data distribution mechanism. They do this by providing some type of API to allow implementers to retrieve data from proprietary data storage and transmit that data back to the client application. Depending on the vendor, and the surface area of their API in relation to the product's functionality, implementers can build solutions as narrow as a plug-in to a vendor's proprietary architecture or as expansive as developing a standalone high-fidelity thin-client desktop application[9,10,11]. None of these offerings directly address data consumption on mobile or personal computing devices. Regardless of extensibility provided through "bolt-on or premium services, all COTS vendors attempt to drive system integrators into a "better together" value proposition that keeps the COTS vendor's products in the mission critical system data flow and leaving customers even further surrounded in a walled garden.

At the time we began our initial research, OMG had just released a Web-enabled DDS RFP. That RFP is still being matured by OMG[12]. DDS in its current implementation did not have implementations available for iOS or Android and would have required substantial upfront engineering to demonstrate our concepts. Further, the required upfront engineering would have only supported two PMCD platforms; our overarching goals were to maximize the number of supported PMCD platforms.

In order to minimize long-term software development costs through maximizing code reuse, we posit the most cost effective approach is to design as many components of LIMO as possible in a COTS agnostic way, reducing or eliminating altogether the need to purchase "bolt-on" or premium services.

C. Proposed Data Flow

Based on our architectural goals, the state of current vendor products, and the maturation of associated protocols/standards, we propose the data flow illustrated in Fig. 4. In this figure we group together three primary responsibilities: Data retrieval/rendering, data conditioning/storage, and data extraction.

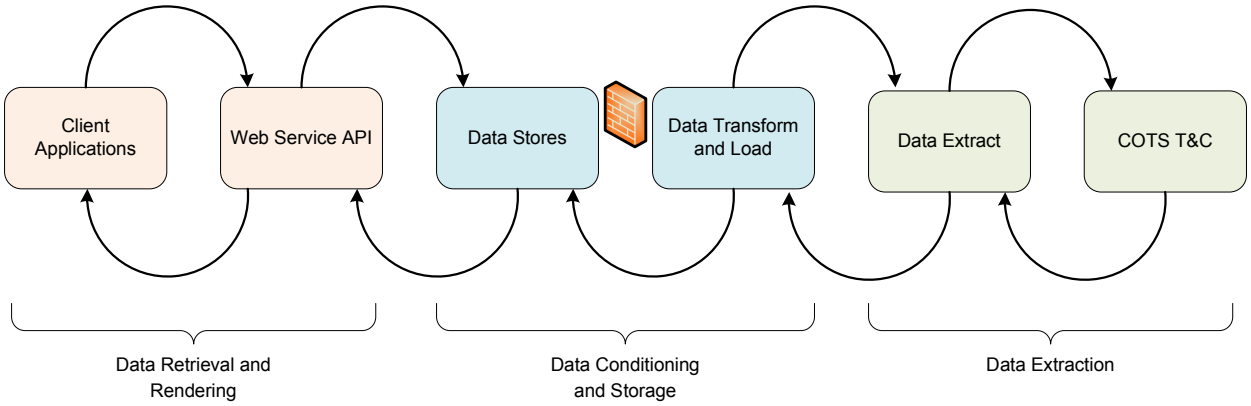

Figure 4 - Data, extraction, storage, and retrieval are the primitives of any LIMO solution

Data retrieval covers the applications resident on the PMCDs along with the mechanism those applications employ to access data from the data cache / data store. Client applications running on various PMCDs render retrieved data in ways that are appropriate for the device.

The data storage layer transforms extracted data into formats necessary for easy search, query, merge, and other conditioning operations performed by DMZ servers. Data loading populates these transformed data into the datastores serving as the cache for future retrieval by PMDCs. In our proposed architecture we provide a one-way mechanism out of the MOC during the data storage phase; data, once transformed and loaded for delivery to PMDCs, is not flowed back to the MOC systems based on current security policies. These data stores expose their data via a Web Service Application Programming Interface (API). This technique provides the widest PMCD platform penetration as it is supported by all devices in our use study.

The interface between the MOC T&C computers and the DMZ servers is enabled through institutionally-controlled firewalls using well administered security protocols. The query mechanisms are similar to those used by other MOC applications that query data from the underlying data archive and real-time data streams.

D. Software Architecture

Our software architecture conforms to the architectural goals established from our systems analysis. When implementing the various tiers in the LIMO solution we began with an analysis of the low-level networking stacks, web architectures, and COTS T&C core software integration necessary to generate and publish NRT telemetry. This included a survey of available security frameworks, development kits, and JHU/APL networking policies. We chose the Representational State Transfer (REST)[13] software model, which requires architecture to support:

- Client-server system operation.
- Stateless operation. There should be no need for the service to keep users' sessions.
- Cache support. The network infrastructure should support cache at different levels.
- Uniform accessibility. Each resource must have a unique address and a valid point of access.
- Layering to support scalability.

These constraints do not dictate what kind of technology to use; they define how data is transferred between components. A RESTful solution can be implemented in any networking architecture available using existing networking infrastructure and protocols[14].

V. Prototype Implementations

Over the course of two years, we implemented a technical demonstration and an operational prototype for NASA's Solar Terrestrial Relations Observatory (STEREO) and the Radiation Belt Storm Probes (RBSP), respectively. We built working prototypes of thin-client applications for the iPad, iPod Touch, and the Android OS.

A. Initial Prototypes

Two working prototypes for PC web browsers were also made to support thin-clients on existing desktop and laptop infrastructure. The first used HTML, CSS, and JavaScript and employed Asynchronous JavaScript and XML (AJAX) techniques. The second used Adobe Flex, an open source framework for building web applications. With the exception of the Flex prototype that used XLM, all other client implementations exchanged resources as JSON objects. The client implementation sub-team had no difficulties finding capable JSON libraries to use across the various platforms (iOS, Android, etc.)

In our earlier technical demonstration, it became apparent that supporting the various native platforms was costly and required many resources from different technical skills. For our more recent operational demonstrations with RBSP, we focused our attention towards our overarching principle of wide PMCD platform adoption. We looked into approaches for keeping the development costs of client implementations down and reduce the total number of lines of source code that must be maintained. We shifted our client implementation strategy to focus on standards-based responsive web design. Responsive Web Design is a term used to describe web applications that adapt to the media that renders them[15]. We took advantage of an existing web application framework, jQuery Mobile, to rapidly prototype a client implementation that could be used on a variety of PMCDs.

B. Mission Operation Integration

To achieve our research objective of demonstrating a NRT data distribution platform, the operational concept had to integrate the various layers of our back-end system and properly interface with our client prototypes, both on the JHU/APL private network and outside the JHU/APL network on the public Internet. This required a web server with a valid SSL certificate to exchange authenticated Hypertext Transfer Protocol Secure (HTTPS) requests and responses. Further, it required the web service API implementation to access data from the database and process the response back to the necessary clients.

We utilized Apache and Apache Tomcat for web server and a web application server, respectively. Apache has been the world's most popular web server since the mid-nineties and Apache Tomcat is an open source software implementation of the Java Servlet and JavaServer Pages technologies. To maximize the development time allocated on this research project, we utilized Jersey, a pure Java RESTful framework. Jersey is the reference implementation of Sun's Java API for RESTful Web Services, also known as JAX-RS. The JAX-RS project was started by the Java Community Process (JCP) with the goal of creating an API for Java RESTful web services.

C. Lessons Learned

Given the short duration of the technical demonstration, there was only a small portion of time dedicated to the various client application HCI. While there is a wave of multi-form factored devices hitting the consumer electronics market, many with rich touch / gestural interfaces our focus was to validate the concept of NRT, secure data distribution. The resultant platform and Web Service API was demonstrated to multiple operational missions, using their mission data, via various mobile devices and two web applications.

It was possible to query, decommutate, and transfer multiple telemetry point values to a data caching layer. During an active pass with the spacecraft, telemetry can be decommutated by the COTS T&C product, processed by the data ingest layer and inserted into the database in the data cache in millisecond times. When accessed by a Web Service client, the telemetry data stored in a MySQL database in the data cache layer are queried out by the classes that implement the web service handlers; again, in millisecond times. By keeping all the various processing and translation times down to milliseconds, the end-to-end time from spacecraft to location independent mission operations users has remained in the sub-second to second time frame.

Users of all the various client prototypes supplied username and password credentials before logging into their applications / browsers. The username and password were passed in the HTTP authentication request header (Base64 encoded) of each call made to the web service. Each call is an HTTPS call and Apache Tomcat was configured to validate the username and password against a table in the MySQL database. Requiring logon credentials, communicating requests over HTTPS, and keeping a data cache for telemetry data, instead of reaching back into the MOC for data further mitigated the potential security risk to mission critical data. Not supporting a mechanism to push data or effectuate a change in the MOC also helped mitigate the risks in providing the Web

Service access to the Internet. Lastly, when users exit / suspend their applications, no data is written to the device's persistent memory.

VI. Conclusion

The continued evolution of mobile computing devices, for the first time in two decades, is changing fundamental aspects of the HCI. Concurrently, ground architectures for new missions review emerging technical means to evaluate their ability to reduce cost and increase effectiveness of applications for mission operations. The personal, mobile computing device presents the ability to provide focused, secure, intuitive data access to mission operators regardless of their geographic context. However, the benefits of these devices are most effectively realized when applications and architectures are designed around their inherent weaknesses (processing power, screen size, input methods, and battery life). Architectures that support multiple devices based on open standards show the most utility and lower adoption cost than dedicated solutions tied to a specific PMCD API. Reference implementations of these capabilities using operational missions at JHU/APL successfully demonstrate the viability of secure, NRT data distribution. We predict that the appropriate incorporation of these devices into ground architectures does disseminate flight data to experts as necessary, enables new opportunities for collaboration, and removes logistic burdens and costs associated with continuous support of missions. Just as PMCDs are replacing laptop computers for mobile data consumption of entertainment media, as ground architectures evolve to support them correctly, we see them becoming an indispensible tool for future mission operations.

References

[1] RUTKOWSKI, C. An introduction to the human applications standard computer interface, Part 1: Theory and principles. BYTE 7, 10 (Oct. 1982), 291-310.

[2] Kent L. Norman, Linda J. Weldon, and Ben Shneiderman. 1986. Cognitive layouts of windows and multiple screens for user interfaces. *Int. J. Man-Mach. Stud.* 25, 2 (August 1986), 229-248. DOI=10.1016/S0020-7373(86)80077-3 http://dx.doi.org/10.1016/S0020-7373(86)80077-3

[3] Hrvoje Benko, Andrew D. Wilson, and Patrick Baudisch. 2006. Precise selection techniques for multi-touch screens. In *Proceedings of the SIGCHI conference on Human Factors in computing systems* (CHI '06), Rebecca Grinter, Thomas Rodden, Paul Aoki, Ed Cutrell, Robin Jeffries, and Gary Olson (Eds.). ACM, New York, NY, USA, 1263-1272.

[4] Jörn Hurtienne, Christian Stößel, Christine Sturm, Alexander Maus, Matthias Rötting, Patrick Langdon, John Clarkson, Physical gestures for abstract concepts: Inclusive design with primary metaphors, Interacting with Computers, Volume 22, Issue 6, November 2010, Pages 475-484, ISSN 0953-5438, 10.1016/j.intcom.2010.08.009.

[5] Asaf Shabtai, Yuval Fledel, Uri Kanonov, Yuval Elovici, Shlomi Dolev, Chanan Glezer, "Google Android: A Comprehensive Security Assessment," IEEE Security and Privacy, vol. 8, no. 2, pp. 35-44, March-April 2010, doi:10.1109/MSP.2010.2

[6] Posegga, J., and Schreckling, D., *Next Generation Mobile Application Security*, Vieweg+Teubner Verlag, 2011, pp. 181-199. ISBN: 978-3-8348-8256-1 Url: http://dx.doi.org/10.1007/978-3-8348-8256-1_10 Doi: 10.1007/978-3-8348-8256-1_10

[7] "Mobile/Tablet Operating System Market Share", http://netmarketshare.com/mobile-market-share (accessed 24 April 2012).

[8] Wroblewski, Luke. "Touch Gesture Reference Guide". http://www.lukew.com/ff/entry.asp?1071 (accessed 24 April 2012).

[9] Harris Corporation. Network Enabled Operations. http://download.harris.com/app/public_download.asp?fid=2093 (accessed April 8, 2012)

[10] Integral System's. Webic http://www.integ.com/ProductBriefPDF/Webic%20Datasheet.pdf (Accessed April 8, 2012)

[11] L-3 Telemetry West. InControl http://www2.l-3com.com/tw/pdf/datasheets/ML_InControl_Brochure_2010.pdf (Accessed April 8, 2012)

[12] Pardo-Castellote, Ph. D., Gerardo. "Web-Enabled DDS: Accessing Real Time DDS Data from Web-Enabled Clients" http://www.omg.org/news/meetings/GOV-WS/pr/rte-pres/DDS_WebEnabled_RTEW09.pdf (accessed April 18, 2012)

[13] Fielding, R, "Architectural Styles and the Design of Network-based Software Architecture" PhD diss., University of California, Irvine, 2000.

[14] Sandoval,Jose. RESTful Java Web Services. PACKT Publishing, 2009.

[15] Marcotte, Ethan. *Responsive web design*. New York: A Book Apart, 2011.