Marrying Social Media Approaches and Space Flight Control: Eight Years at SpaceOps

David.W. Scott[1], C.M. Albers[2] Ph.D. PMI-ACP, H.S. Cowart[3], A.J. Nichols[4], and R.L. Roy[5]
Payload Mission Operations Division, NASA - Marshall Space Flight Center, ,
Huntsville, AL, 35812, USA

Three previous SpaceOps papers [1-3] - published in 2010, 2012 (honored by the Conference as a "Best Paper"), and 2014 - have discussed paths to using social media concepts and techniques to enhance space flight controller effectiveness by a) reducing clutter of non-verbal communications (e.g., visual flow with minimal headers and shared content instead of multiple copies), b) moving some voice communication to non-verbal transmission (virtually eliminating "say again" requests because non-verbal comm can be re-read), thus making remaining voice comm easier to focus on, and c) reducing short-term and long-term flight stress on flight control personnel. This paper shows how Marshall Space Flight Center's (MSFC) ISS Payload Operations Integration Center (POIC) is realizing the above goals via the Communications Dashboard (CommDash) software suite deployed in 2017 (including enhancements to the Console Log Tool (CoLT) discussed in earlier papers). Two larger-scope benefits spawned by CommDash evolution are also chronicled: a) emergence of an Agile Software Development (ASD) process adapted to the not-always-nimble environment of government projects, and b) the sprouting of a Human Factors Engineering (HF or HFE) community of practice within MSFC's Payload and Mission Operations Division (PMOD).

I. Introduction

Space flight control involves a river of decisions, actions, and observations, often documented or tracked in multiple applications (leading to data replication fatigue), particularly if multiple control centers are involved. Estuaries that flow into the river include telemetry, voice reports internal and external to a control center, planning systems and reports for human activity, data flow, commanding, and person-to-person exchanges, to name just a few. It should come as no surprise that flight control personnel and teams may be saturated by the total flow and sheer volume of visual, audible, and written traffic, especially during contingency situations when time is of the essence.

After the final Space Shuttle flight to the International Space Station (ISS) in 2011, ISS began migrating from Assembly operations to Utilization (payload operations), and the amount of ground coordination for payloads naturally began to increase. Development of U.S. based Commercial Crew Vehicles (CCV) raised the potential for having 7 permanent crew members onboard ISS instead of 6, e.g., have a 3-person Soyuz and a 4-person CCV docked as lifeboats. NASA successfully campaigned for having the extra crew member dedicated to NASA payload operations (which includes any payload flown in a NASA rack/facility, regardless of physical location or the payload sponsor). Since 3 members of a 6-person crew were already assigned to NASA payload ops, the term "4th crew" was coined. Eventually, the term evolved to High Operations Tempo (HOT). The two terms are equivalent.

In the 2012 time frame, ISS averaged 25 hours of crew-tended NASA payload ops per week. The goal for 4th Crew (originally anticipated to start in 2017) was set at 100 hours per week. However, projected increase of payload planning and execution complexity was more than linear with respect to the increase in crew hours. Simply hiring, training, and maintaining hordes of flight controllers and using 2012-era Concepts of Operations (ConOps) would have been cost

[1] Systems Engineer, NASA HP27, scotty@nasa.gov
[2] Software Build Coordinator and Agile Consultant, COLSA Corpration, cerese.m.albers@nasa.gov
[3] Project IntegratorTeledyne Brown Engineering, hugh.s.cowart@nasa.gov
[4] Payload Operations Director, PMOD, andrew.j.nichols@nasa.gov
[5] Systems Engineer, COLSA Corporation, Huntsville, AL, 35806, USA, robert.l.roy@nasa.gov

prohibitive, so in mid-2014, MSFC's Mission Operations Laboratory (MOL) launched a major initiative to review and evolve ConOps, Flight Control Team (FCT) composition, ground procedures, and automation/tools. (Note - In early 2017, most of the MOL, which had been part of MSFC's Engineering Directorate (ED) for the better part of several decades, was reorganized as the Payload Mission Operations Division (PMOD) within a new, ED-independent Human Exploration Development and Operations Office (HEDO or HEDOO).

In 2012, a handful of MOL engineers had begun pondering the idea of adapting principles, techniques, and rather intuitive interface characteristics associated with social media to the payload operations environment, and proposed concepts and high level requirements for a POIC-specific Communications Dashboard (CommDash). In addition to suggesting innovative ways to organize, manage, and display information, they acquired tools for user interface prototyping and asserted that increasing the practice of Human Factors engineering, particularly for ground system displays, might significantly improve tool leverage and reduce user stress.

Some of the application proposals adopted for 4th Crew involved automating well-established processes, functions, and data, and could be built efficiently via NASA's traditional "Waterfall" development cycle. However, POI also selected CommDash and some other evolutionary tools for development. These had functions and ConOps that were very new, at least to POI and the Huntsville Operations Support Center (HOSC) (which houses and provides infrastructure for POIC and other mission-related tenants), so a much more flexible development scheme was needed.

The ISS Program, via the POIC Project, funded creation of an Agile Software Development (ASD) process specific to POIC needs, e.g., adopt Agile principles and techniques yet respect constraints on operations personnel availability, NASA project budget practices, and ISS flight increment-based delivery schedules. ASD made it possible to make discoveries and requirements/implementation changes early on, with very little recoding needed during IV&V or acceptance. The process may be applied to a range of projects that PMOD is involved with or may support in the future, hence the term PMOD ASD.

II. Early Papers on Using Social Media for Serious Work, CommDash Origins

Circa 2009, one of this paper's authors, David Scott, who has been a payload crew communicator for Spacelab and ISS (Expeditions 3-15) missions, noted that Facebook®, despite its reputation for being frivolous from a professional standpoint (e.g., Farmville, other games, and advertising), did a very good job of visually aggregating several streams of information and hiding, emphasizing, and showing certain times of content under appropriate conditions and in empathy with the user. In 2010, Mr. Scott wrote two papers [1, 4] on how Web 2.0 and social mechanisms such as those in Facebook®, Twitter®, and the like might be harnessed to support engineering support and/or operations activities. He collaborated with another author of this paper, Hugh Cowart, and Dan Stevens of Johnson Space Center (JSC), on a Space Ops 2012 paper about a newly deployed Console Log Tool (CoLT), Communications Dashboard concepts for the POIC environment, and tag-based, inter-log communications among JSC's Microsoft Word-based console logs related to the earlier papers. [2]

Based on ideas spawned while writing the 2012 paper, Scott sought and received funding for a NASA Headquarters IT Labs (Information Technology) Idea Phase study on ways to create a test bed for using a social networking service (SNS) approach to improve a real time control center operator's ability to keep up with both the 'big picture' and critical minutiae of operational events and context. The essence of the proposal (and a micro-synopsis of the early papers) may be grasped in 60 seconds via the proposal's "Elevator Pitch".

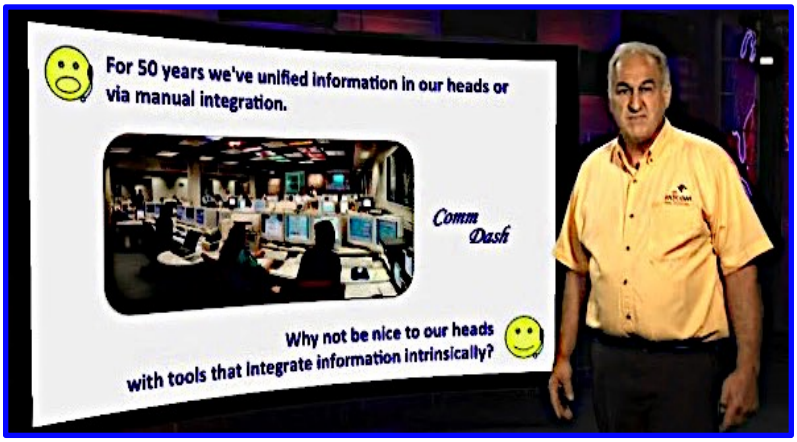


Fig. 1 Motivation for pursuing Communications Dashboard [Video]

Primary accomplishments of the FY13 effort included

- Concept and high-level implementation discussions with MSFC and JSC operations personnel, Huntsville Operations Support Center (HSOC) staff, and systems engineering and expert systems personnel at Jet Propulsion Laboratory (JPCL, Ames Research Center (ARC), and commercial vendors.
- Established working relationships with a U.S. Navy systems organization that successfully implemented text chat for operational communications in the early-2000's and with an Army Research Lab detachment specializing in visualization for Human-Computer Interface displays.
- Acquisition and experimentation with display prototyping tools.
- Draft of salient features to develop or mashup.
- White Paper – "Communications Dashboard (Control Rooms, Take a Cue from Facebook®!) Chapter 1" [5]

While early work revolved around a Facebook®-like appearance, a 2014 CommDash test bed proposal included high level requirements that intentionally avoided detailed visual format specifications, to avoid bias that could blind us to useful, non-Facebook® ideas. Though the test bed was not approved for development, payload operations personnel selected CommDash as their first priority for development as a HOT tool about a year later, and the test bed Level A document evolved into production requirements. The departure from Facebook®-based appearance enabled the product team to generate features uniquely suited to the POIC environment.

III. The Process - Hybridized Agile Software Development

A. Generic Overview of "Agile"

A surprisingly large number of people discuss "Agile" without knowing what it really is or its origins. In 2001, seventeen leading proponents of non-traditional approaches to software development (some were both competitors and collaborators with respect to each other) met for three days and produced a 68-word philosophy statement known as

The Agile Manifesto

We are uncovering better ways of developing software by doing it and helping others do it.
Through this work we have come to value:

Individuals and interactions over processes and tools
Working software over comprehensive documentation
Customer collaboration over contract negotiation
Responding to change over following a plan

That is, while there is value in the items on the right, we value the items on the left more.

Major concepts underlying the Manifesto are laid out in 186 words at http://agilemanifesto.org/principles.html, and the rest of the parent site is well worth perusing. While Agile is more of a mindset than a method, many Agile approaches involve some variant of Scrum (named after the huddle that occurs in rugby football), and a simple web search on "Scrum" is also worth a try. Scrum's founders are 2 of the 17 Manifesto authors. Agile approaches are not limited to software development, and have much in common with lean manufacturing.

A 113-word Declaration of Interdependence (http://pmdoi.org/), originated in 2005 by fifteen authors, one of whom helped to write the Agile Manifesto, defines six values that provide "a modern view of managing projects, particularly the complex, uncertain ones," in the interest of promoting "Agile and adaptive approaches for linking people, projects and value."

A. Overview of PMOD Agile Implementation

MSFC has traditionally used a sequential, Waterfall Software Development (WSD) process that focuses on a Plan-Do-Control flow of Requirements refinement upfront, Detailed Design, Implementation, Testing, and Deployment. WSD works well for clearly-defined projects. The goal is to complete the project according to requirements, on time, within the planned scope and cost.

In contrast, Agile Software Development (ASD) preeminently emphasizes value. The goal is to deliver as much quality as possible by letting the customer *discover* the real requirements by exercising iteratively delivered, discrete pieces of the product, providing feedback to, and negotiating changes with, the developers, and re-evaluating/re-negotiating the product as it grows to maturity. Later sections of this paper explain how PMOD evolved an Agile-inspired process while honoring some not-terribly-flexible realities of the flight operations environment.

Fig. 2 characterizes Waterfall and Agile approaches and the scenarios for which they are suited.

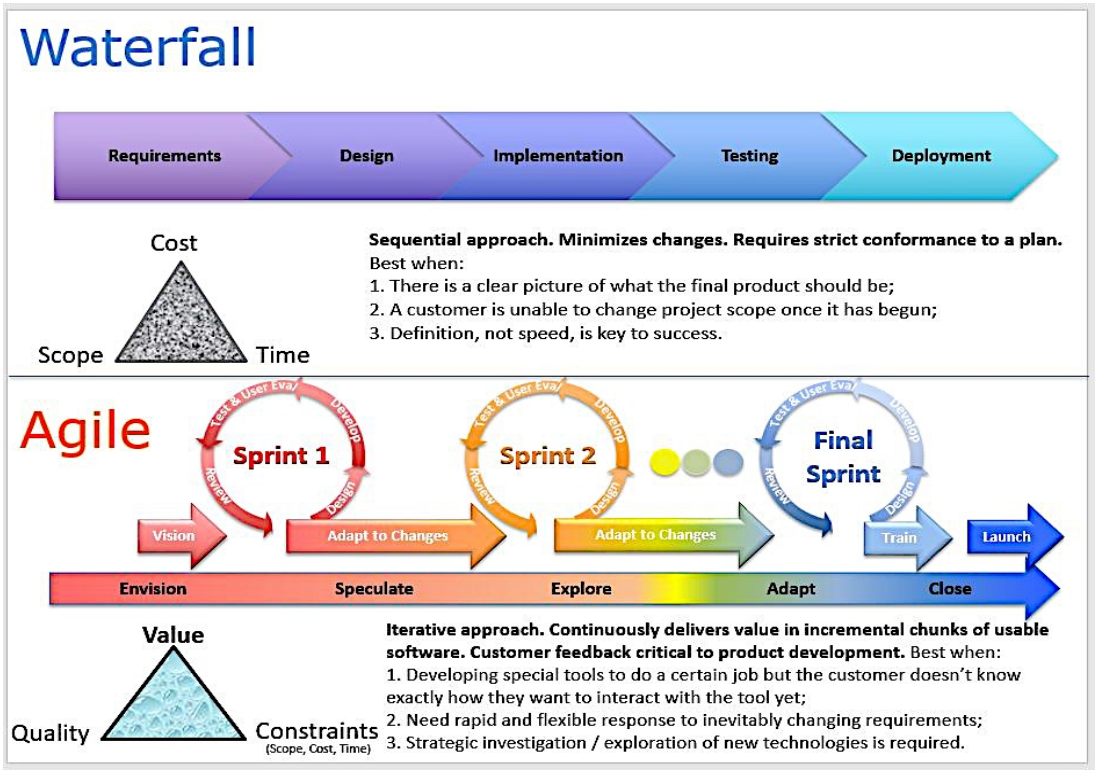
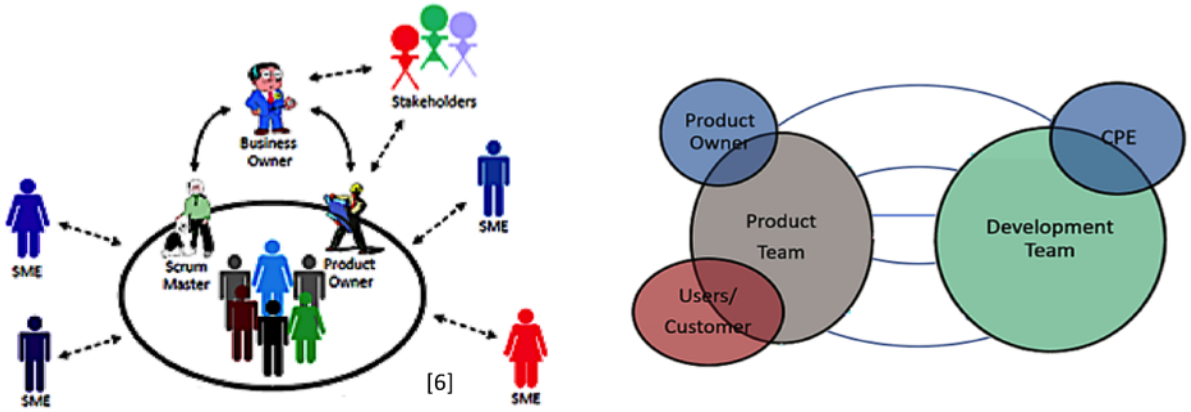


Fig. 2 Waterfall and Agile approaches comparison

There are two fundamental differences between typical ASD and PMOD ASD:

- Agile development for a project (or sub-project of a complex effort) usually consists of a single Product Development Team that includes developers, stakeholder representatives, a Product Owner representing the Customer, sometimes a few end users, Subject Matter Experts (SMEs) as needed, and a Scrum Master or Moderator to guide execution of the method/process. The team meets daily for 5 to 10 minutes maximum to tag up on what happened the previous day and what will be accomplished on the current day. This daily standup meeting also collaboratively identifies obstacles that need to be removed, and calls out anticipated delays, challenges, or resource issues that need to be worked out. Details are then discussed offline. In the PMOD environment, this is simply not practical for operations personnel due to console shift schedules, training requirements, and other meeting commitments. For PMOD ASD, we established a Product Team of operations personnel, moderated by a Product Owner from operations, and a Development Team of software developers, testers, trainers and security personnel, moderated by a Change Package Engineer (CPE) from the HOSC, as illustrated in Fig. 3.



Typical Agile: Single Product Development Team

PMOD Agile: Separate Yet Well-Coupled Product and Development Teams

Fig. 3 Distinction between Typical Agile and PMOD Agile Team Structure

- Agile teams usually have a fair amount of leeway with respect to schedule and, to some extent, cost. PMOD software deliveries related to ISS are tied to flight increment (expedition) configuration dates, NASA financial conventions, and PMOD control board practices, so our ASD approach is a time-boxed, hybridized Agile Software Development paradigm. Flexibility is provided in the way the work is approved and may proceed with reasonable accommodations. As with most hybrid Agile methodologies, a choice of which of the three primary project constraints will be held fixed is made and the rest are more flexible. In the PMOD hybridized Agile implementation, time is the most fixed, followed by cost which can vary some, and the most variability is permissible with project scope.

This would be a good place to point out how this paper's authors relate to the bubble diagram from the right side of Fig. 3:

- Dave Scott, alias Scotty, had originated the CommDash concept circa 2011 and served as an (enthusiastic) advisor, idea generator, and Human Factors advocate, primarily on the Product Team side.
- Dr. Cerese Albers architected our custom ADS approach and shepherded engineers and managers of all stripes through the learning curve on how to make the process our servant (and not the other way around).
- Hugh Cowart was instrumental in identifying and defining innovative features for the original (2011) CoLT log keeping tool, and continued the practice for CommDash.
- "Jay" Nichols, an ISS Payload Operations Director, was and is the CommDash Product Owner.
- Rob Roy, a Systems Engineer at the HOSC, was and is the CommDash CPE.

Dr. Albers has written another SpaceOps 2018 paper, "Hybridized Agile Software Development of Flight Control Team Tools for International Space Station's Payload Operations Integration Center" containing in-depth discussion of PMOD ASD structure and management.

B. Lessons Learned From Agile Development of CommDash

CommDash was the second project developed using PMOD ASD, and is the largest and most distributed PMOD ASD project to date. As such, it became a proving ground for scaling the PMOD ASD paradigm from a small project to a larger one and yielded many pivotal lessons learned for process improvements.

Major challenges to implementing CommDash included:

- Assessing resources needed to provide an accurate ISS Increment delivery timeframe - CommDash had origins in trade studies, one of which was included in an early ConOps. There was a plethora of data and ideas, yet not enough information on how CommDash would actually be used in real time operations, both as a whole and for each component application. In the course of clarifying this, standard information to be included in all ConOps documents emerged:
  - Use cases for both Flight and Simulation scenarios
  - A Table listing of Minimum Success Criteria (MSC) for the end product
    - MSC capabilities - must be met for Product Team acceptance
    - "Highly Desired But Not Required" capabilities - for inclusion if time and resources permit
    - "Nice-to-Have" capabilities – to keep in mind for the future or if opportunity arises somehow.
  - Capabilities must be described at a high level within ConOps, and cross-referenced in the MSC Table
- Scheduling User Evals to promote adequate participation and consistent feedback
  - Stakeholder cadre positions have demanding console and training schedules
  - Even if each position was represented, individual evaluators often changed between sprints, leading to inconsistent inputs.
    - There was no protocol for information handover between sprints
    - New evaluators made change requests that should have been identified in prior sprints, making meetings longer.
  - When a tool was released from a particular phase, varying opinions emerged from cadre members who were not involved in User Evaluations

  Mitigations to scheduling challenges include
  - Project Team identified primary and secondary evaluators and tried to make changes only between phases so that at least all sprints within a given phase had the same evaluator(s)
  - Primary and secondary evaluators took the same Agile training at the start of a product phase and made the commitment to confer after each sprint. This reduced redundancy, reworks, and technical debt, plus facilitated discussion and generated feedback.
  - Evaluators were chosen for schedule availability and for ability to serve as an informed representative of their console position. This reduced variances from non-evaluators.

Some benefits of the ASD process appeared immediately. Others took a while time to come forth but were equally valuable, if not more so. Even early failures were leveraged to create future positive outcomes, and the CommDash team learned together as a whole. As a result, templates, standard feedback mechanisms, and standard operating procedures (SOP) emerged that brought users and software developers orders of magnitude closer in collaboration than they had been in the past. For example:

- SOP for coordinating how to provide new user privileges in a timely manner for different situations (user evaluations, simulations, ground transitions to operations).
- Method and templates for gathering quantitative and qualitative feedback during Sprint activities
  - Supported synchronous and asynchronous inputs during and after User Evaluation sessions
  - Gave Product Owner feedback in meaningful metrics, with a list of comments/questions for building consensus as an internal Product Team
  - Became a centerpiece for feedback delivery to the Development Team at the end of a User Eval and for sprint deliverable acceptance discussions
  - Template for Product Team Feedback form for each sprint became a living communication tool between the teams at the end of each Sprint, until it was signed off by the Product Owner
  - Used a NASA-hosted instance of a collaborative brainstorming and organizing tool (GroupSystems™ ThinkTank®)
- Other templates allowed management to look across the list of cadre tools in work, understand who and what was involved, and know where each tool was in the development process. All information was kept centrally so that project team members could find what they needed.
- Scaled-down User-evaluation SIMulation (USIM) for early interaction with the tool in the evaluation environment

- Product Team (users) and Development Team (testers and software developers) assess the tool working side-by-side
- Typically two-to-three hours in length
- First chance for users to interact with each other in a real-time setting, well in advance of ops sim or flight opportunities.

The CommDash buildup also netted significant intangible gains. Tighter coordination and collaboration between Product and Development Teams gave fresh, significantly deeper insight into each other's constraints and strengths. Empathy for each other's work enriched relationships and caused each side to consider requests more carefully and to be more understanding about time commitments. New forms of collaboration within each team resulted in more fruitful meetings and better sharing of ideas, and sometimes spawned beneficial cadre tool ideas that would leverage existing capabilities. Sub-teams grew in mutual understanding and appreciation by glimpsing how a single tool can affect so many ways of doing business.

The PMOD ASD paradigm celebrates milestones and achievements, and during Sprint Retrospective gatherings, team members were lauded for their commitment to project success. Retrospectives also included fun team-building exercises and informal discussions, making it easier to harvest lessons learned. Giving the teams a chance to relax and have a casually nice time to recognize accomplishments increased the overall affinity for engaging with other teammates in meaningful ways.

Prior to CommDash, individual POIC cadre position teams often pursued consolidation of information sources relevant to that team. While this led to effective solutions for them, it also resulted in a) much duplication of effort and b) parallel, independent information streams containing different bits and pieces of the puzzle. The bits and pieces often did not come together until just before, during, or after (ouch!) the affected operation. CommDash development helped the cadre identify and really understand which information should be visible across disciplines, and the CommDash suite wrangles that information so that:

- Operational decisions can be made in a more timely manner
- More decisions can be successfully managed

This has been vital to keeping up with HOT operations.

IV. The Product - Communications Dashboard Suite

A. Overview of Baseline Implementation

Highlights of CommDash software applications and their associated ConOps include:

- **Dashboard** launches and manages CommDash tools and displays many of them in a single window with user-configurable frames. For ergonomic reasons, some functions require or have options for separate windows. Some applications may also be run completely independent of Dashboard.
- **IMReady** (I Am Ready) is a system for polling the front room cadre positions on simple questions or statuses, and supports multiple concurrent polls. All polls are issued by the Payload Operations Director (POD) position. IMReady was derived from JSC's IMGO (I'm GO) tool (Green-Amber-Red status indicator for each position).
- **Payload Developer Status (PD Status)** summarizes, at a glance, Payload Developer (PD) team and voice-data-video system status and readiness to support POIC-managed activities for the current time, plus and minus 8 hours. The tool autonomously harvests timeline planned activity information from JSC's Operations Planning Timeline Integration System (OPTIMIS) tool, then identifies and groups current and upcoming payload activities. Many fields may be edited via structured dialog boxes with drop-down and/or check box menus for quick selection of valid parameter values. PD Status acts as a sort of status trading post for cadre positions and PD teams. The application also provides display output specifically designed for POIC's video wall; the layout and content is similar to desktop displays yet is optimized for the scale and format of the video wall.
- **To Do List** can be used to define/assign tasks not managed by other systems, identify and hyperlink to tasks a console position is working in another system (Flight Notes, Timeline Reviews, etc.), store/display comments, and show completion status. Front-room and back/prep-room awareness of everyone's tasks improves communications timing and [6]supervisors' effectiveness.
- **Text Chat** supports position-specific channels (analogous to dedicated voice loops), created-on-the-fly streams for specific conversations, and push/pull content to/from CoLT entries. Users may work concurrently in multiple chat streams. Additionally, the chat window itself is configurable, floats above other applications on the dashboard, and provides the user with notifications of new messages and how many are unread. There is also a position-specific chat history that populates when the chat stream opens to give the user conversational context upon logging in. Custom chat rooms may be created and utilized across positions as well.
- **Flight Control Team Log (FCTL)** is a unique CoLT log shared among all cadre positions in front and back rooms. A cadre member may:
  - Push an entry (or part of one) from their position-specific log into FCTL so that all others may see it. (Content in FCTL is a static "snapshot," i.e., it remains the same even if the position log's entry is edited.)
  - Pull an FCTL log entry into their position-specific log.
  - Within the Dashboard environment, use a three-position slider switch to choose viewing options:
    - Position log only
    - Position log and FCTL visually merged, indexed by time, searchable as if it were a single log
    - FCTL only

  This is perhaps analogous to a two-way Twitter feed with a single hashtag - #mutual-interest.

While the tools are useful individually, using them in combination to accomplish a given task can be quite powerful, and is made easier by having them in close proximity to each other in the Dashboard. For example, a plan review might occur as follows:

- To Do list task created to review a plan
- Chat used between or among individual console positions for an off-the-loops discussion
- To Do list task closed out with comments
- Timeline Change Officer (TCO) creates a TCO log entry with appropriate details of review and resulting documentation
- TCO pushes log entry to FCTL, where and other positions can view it and pull it into their logs if desired

B. A Visual Tour of CommDash Functions, Features, and Behaviors

The following annotated figures should provide a general feel for the CommDash environment and the sorts of things it can do. Screen shots were taken from demo and/or test/eval sessions, and do not reflect actual or even simulated payload operations content.

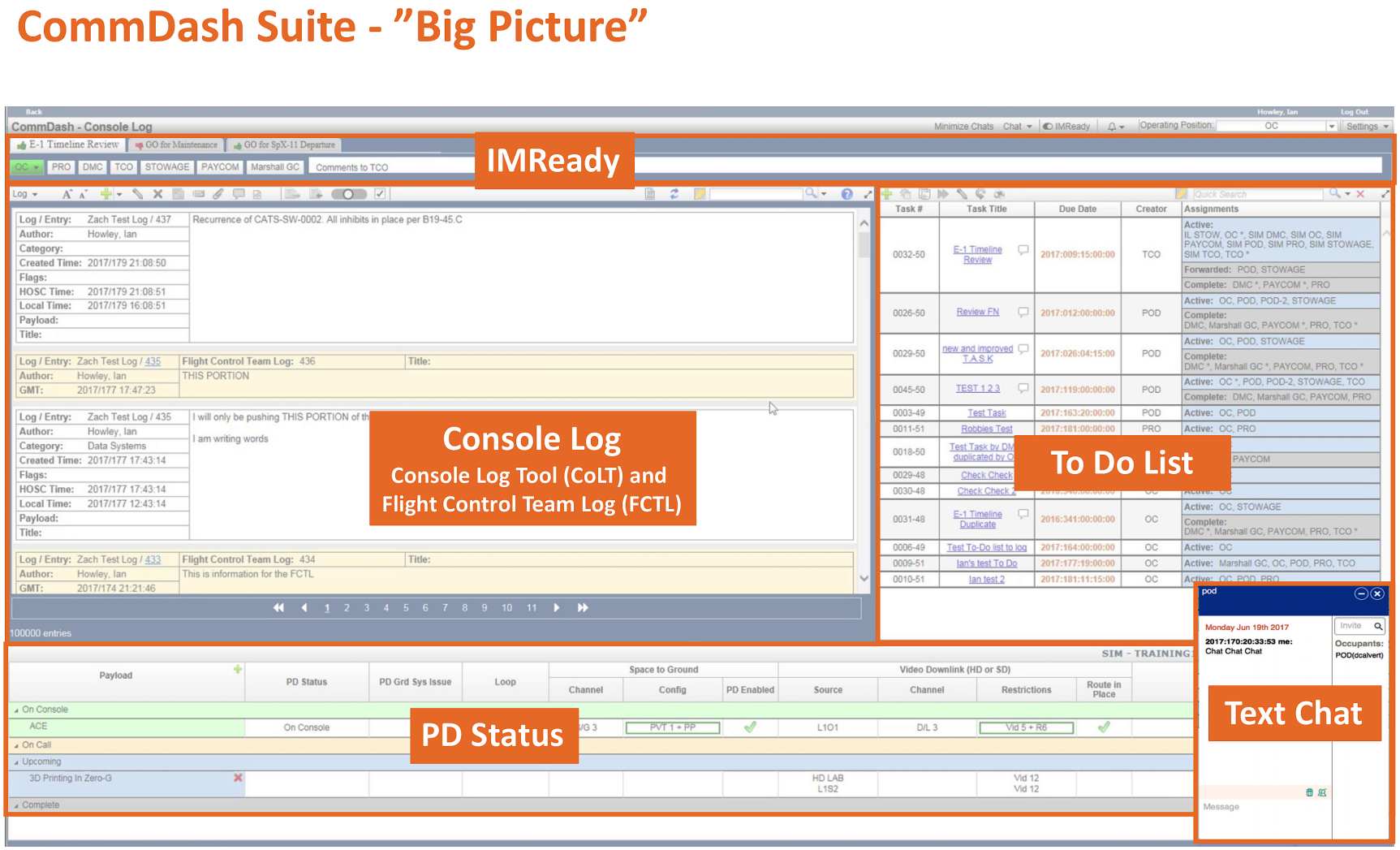


Fig. 4 Typical CommDash display layout, annotated to show frames of the 5 main CommDash applications

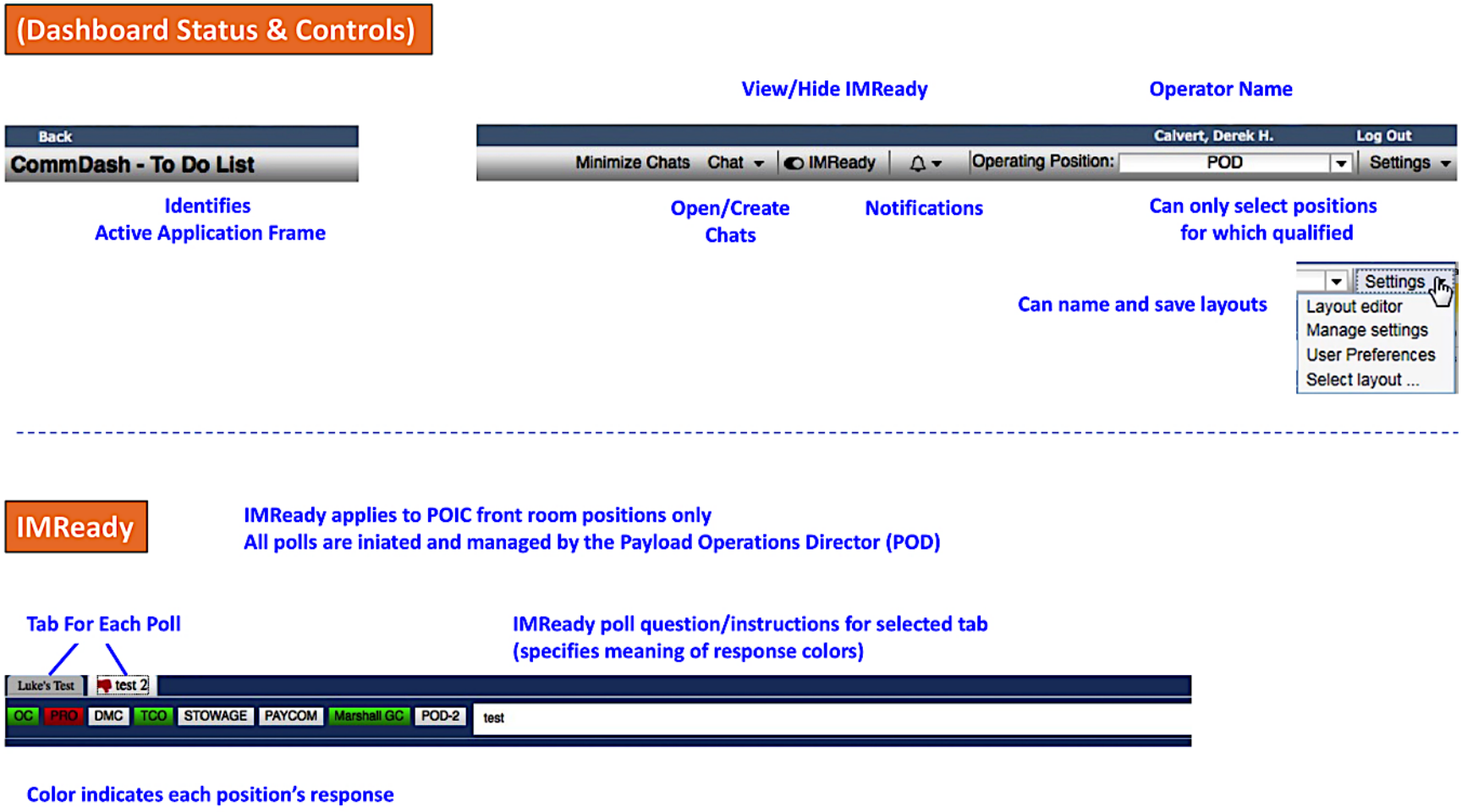


Fig. 5 Details of Dashboard status/controls and IMReady application frem top of CommDash window

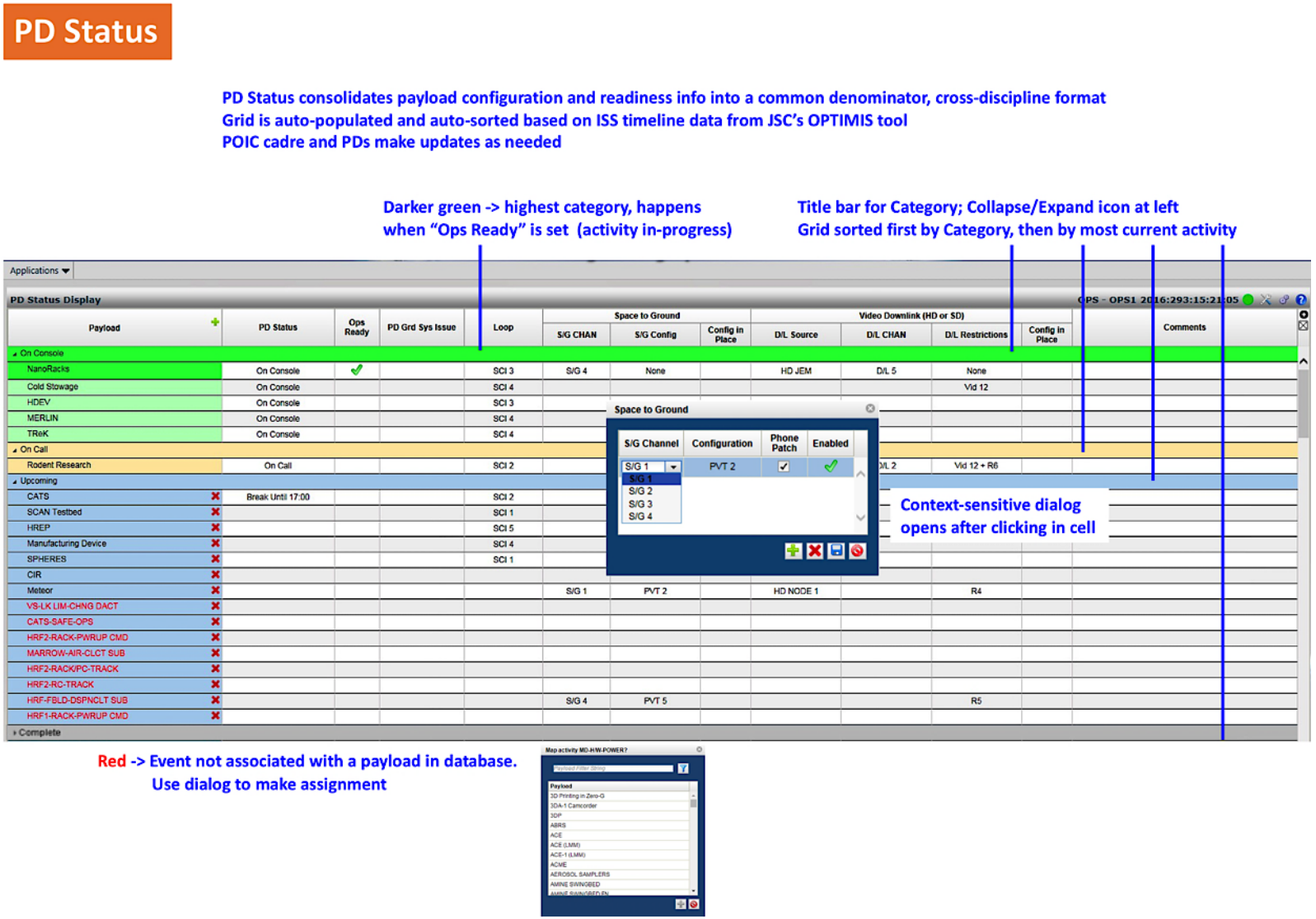


Fig. 6 Highlights of the PD Status application

To Do List

Quick Search

Task #	Task Title	Due Date	Creator	Assignments
0048-50	ALTAIR Ground pass	2017:153:23:00:00	Marshall GC	**Active:** POD
				Complete: DMC *, Marshall GC *, OC *, PAYCOM, PRO
0007-49	HOSC Power Work	2017:167:00:00:00	Marshall GC	**Active:** IPOD, Marshall GC, POD, PREP TCO, TCO
0020-48	Review FN 1234	2016:342:00:00:00	POD	**Active:** SIM STOWAGE
				Forwarded: OC *, STOWAGE *
				Complete: DMC, Marshall GC *, POD *, PRO *, TCO *
0007-51	Test Task	2016:349:00:00:00	POD	**Active:** POD *, POD-2, SIM OC, STOWAGE
				Forwarded: TCO
				Complete: DMC *, Marshall GC, OC *, PAYCOM, PRO *
0026-50	Review FN	2017:012:00:00:00	POD	**Active:** OC, POD, POD-2, STOWAGE
				Complete: DMC, Marshall GC, PAYCOM *, PRO, TCO *
0029-50	new and improved T.A.S.K	2017:026:04:15:00	POD	**Active:** OC, POD, STOWAGE
				Complete: DMC *, Marshall GC *, PAYCOM, PRO, TCO *
0045-50	TEST 1 2 3	2017:119:00:00:00	POD	**Active:** OC, POD, POD-2, STOWAGE, TCO
				Complete: DMC, Marshall GC, PAYCOM, PRO
0047-50	Comm Change	2017:153:23:00:00	POD	**Active:** STOWAGE, TCO
				Complete: DMC *, Marshall GC *, OC *, PAYCOM, POD, PRO *
0003-49	Test Task	2017:163:20:00:00	POD	**Active:** OC, POD
0004-49	Test Task	2017:164:08:00:00	POD	**Active:** POD
				Complete: OC

- Allows any position to assign an action to any other position. Includes the ability to hyperlink products, such as a Flight Note
- Allows assigned positions to status action (Complete/Not) along with plain text commenting.
- Searchable history to locate past tasks
- Can assign due dates for tasks as well as release dates for tasks to be released in the future.
- Documents the name of person and time of completion when task is marked complete

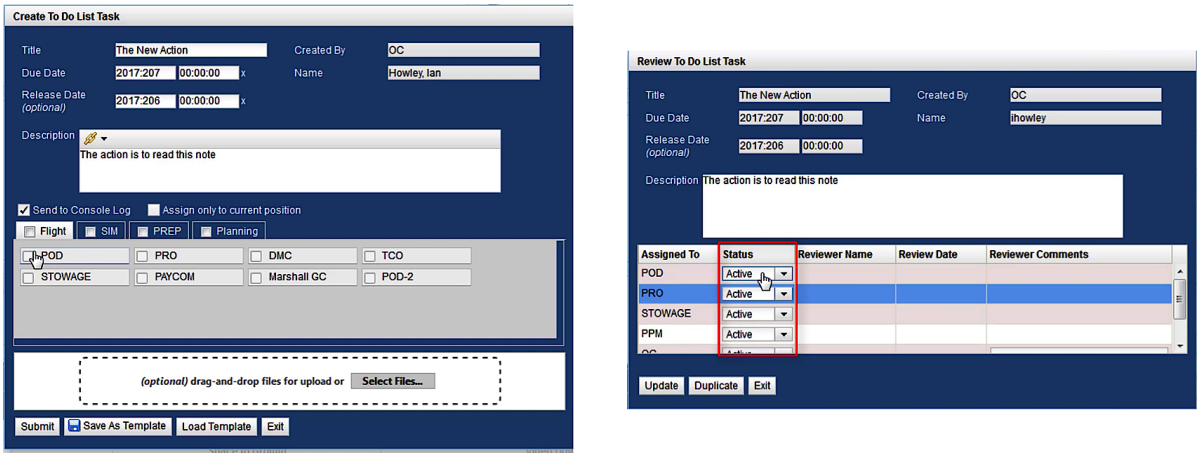


Fig. 7 Highlights of the To Do List application

Text Chat

Chat rooms float on top or can be minimized. Can have multiple chats open. Convention is to keep your position's room open (analogous to keeping your home voice loop on). Content archived.

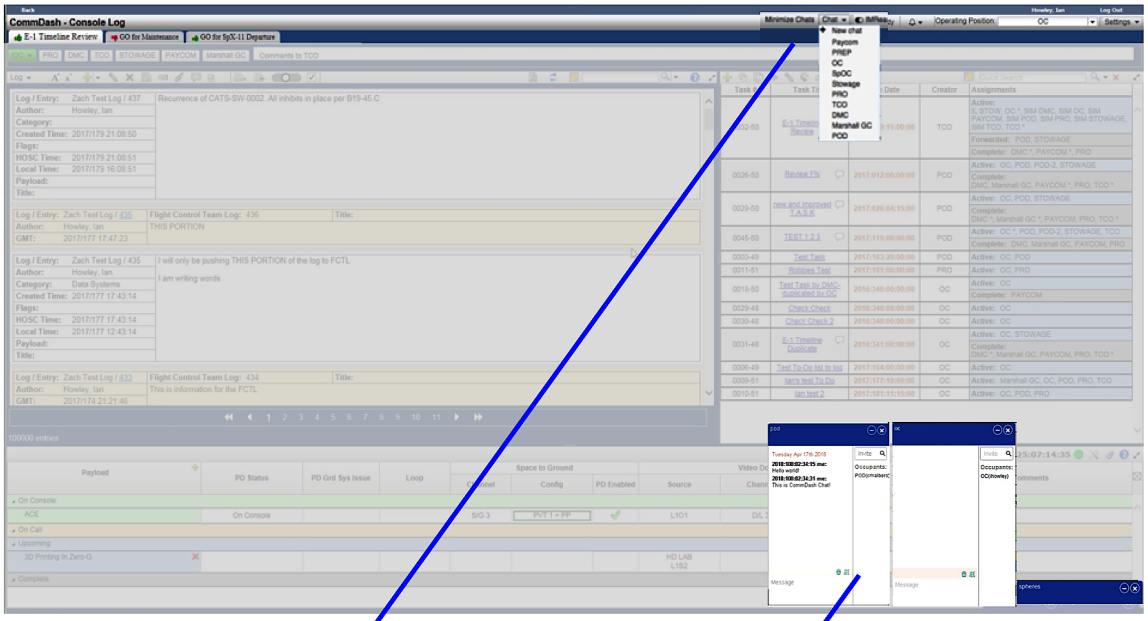


Fig. 8 Highlights of the Text Chat application

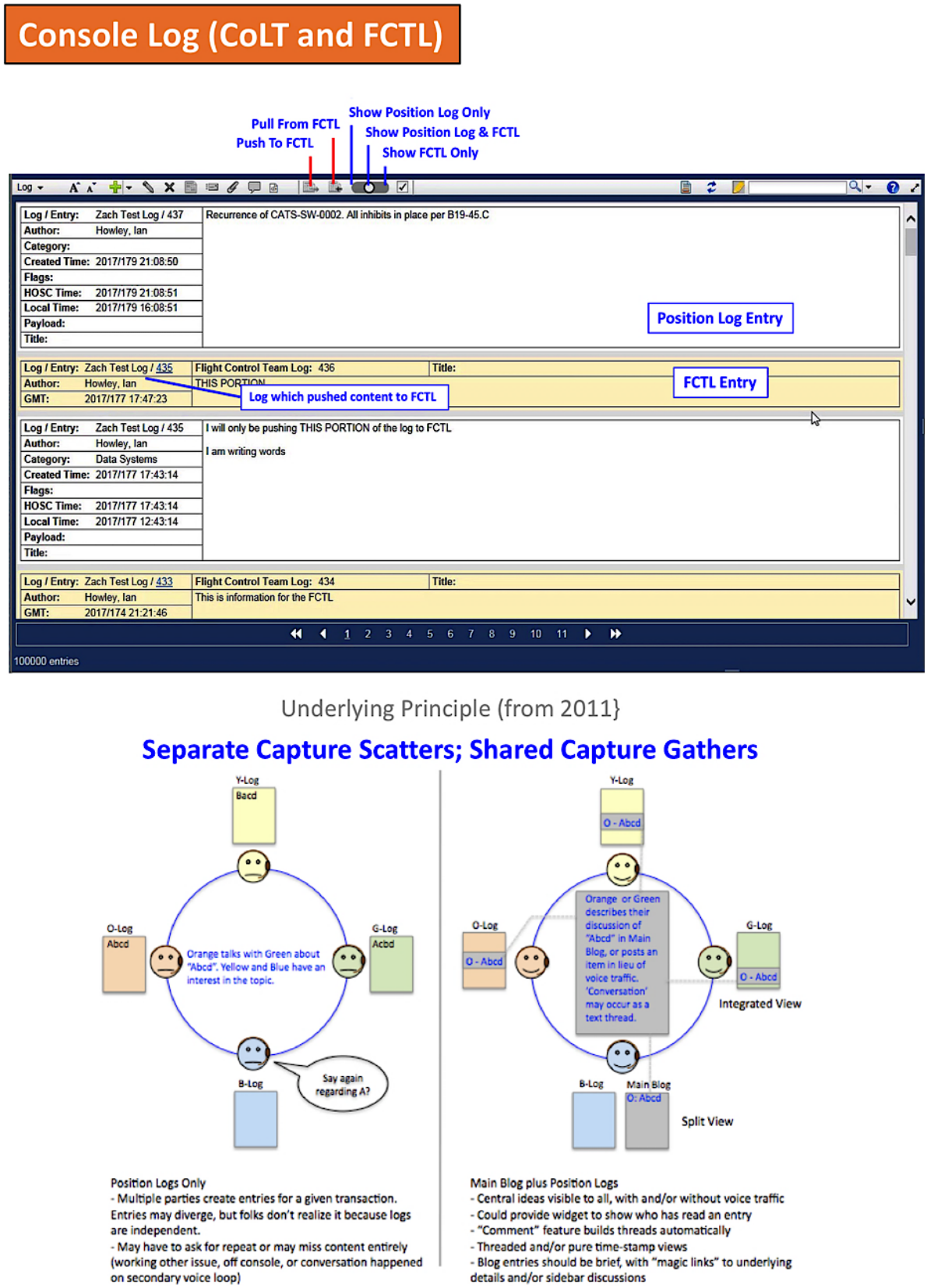


Fig. 9 Console Log functions in CommDash, with correlation to Social Media principles

C. Deployment and Initial Results

The full CommDash suite was released for flight control operations in September 2017, ahead of several other tools created by the HOT initiative. Console positions in both the operations and preparation teams (FCT/Front Room) and "Prep"/Back Room) transitioned to using CommDash incrementally tailoring configurations and use patterns to unique needs and desires, much in the way that water levels itself.

The 4th Crew/HOT initiative to prepare for the payload activity level that has become a nominal pace has officially ended, so CommDash and other tools created or upgraded for current ops are now referred to simply as "cadre tools."

We have analyzed CommDash's operational use in order to gain insight on the tool, the ConOps on which we based the tool, and the software and ops development processes used to get us to this point. Some items we're keeping our eyes on include:

- To what extent should CommDash use be standardized among the cadre (for consistency and clarity) vs. customized by console position teams or individuals. A fluid operational environment such as POIC can benefit significantly from a flexible-yet-measured approach to using multiple communication tools. At times, POIC has been engaged in four concurrent crew activities involving four Space Station crew members communicating on three space to ground channels.
    - At present, cadre are relatively free to choose how they'd like to use the tools. Training materials state, "Different cadre teams may use the tools in a slightly different way. That's OK! Do what makes sense! The key is to understand the underlying strengths/weaknesses of individual tools then use as you see fit. Uniformity isn't the goal, efficiency is."
    - Consensus on a given way to use one or more tools across positions for a specific purpose may be documented in the Payload Operations Handbook (POH).
    - One principle and protocol remains fixed: All critical communications shall take place over the established voice loop channels.
- Are there any "low hanging fruit" Human Factors considerations that might significantly improve flight operations?
- It might be wise to examine the CommDash ConOps occasionally and, in light of accumulated experience with the tool, define or refine a) basic, common functional requirements needed across all control room positions, b) which positions or users have special privileges or capabilities based on their unique function or need, and c) use cases detailing how CommDash might be used by different console positions in various circumstances. At the very least, swapping stories and ideas (logical and otherwise) may protect against the "we've always done it that way" syndrome.
- In current tool releases, much of the input is by humans. (The tools do a nice job managing and manipulating that input in useful ways.) As operations mature and resources for updates become increasingly available, increased content generation from ground systems (e.g., Exception Monitoring (EM) messages via operator selection, EM prompt with operator confirm/cancel, or automatic for pre-defined situations) is a distinct possibility. Future work of this kind will probably use the ASD paradigm because the types of automation and user/computer interactions anticipated will cross very new territory, and efficacy and operations process/data integrity will be at a premium.

Table 1 summarizes results from the first six months of CommDash operational use, based on server statistics, surveys of FCT personnel, and follow-up discussions.

Table 1 Insights from First Six Months of CommDash Operational Use (Sep 2017 – Mar 2018)

Legend for Cadre Position References in 3rd Column

Assistant Payload Operations Manager	APOM (Backroom)
Data Management Coordinator	DMC
Operations Controller	OC
Payload Communicator	PAYCOM
Payload Operations Director	POD
Timeline Change Officer	TCO

Component	Description/Function	Primary Users, Characteristics
CommDash Suite	All CommDash components and behaviors as an environment	• Voice loop clutter has been reduced because CommDash provides places for traffic that can be worked at a slower pace. Since less stuff comes in to POI by voice, it no longer feels like everything is clamoring for priority.
Dashboard	Executive (Launcher, Multi-Frame Display)	
Console Log Tool (CoLT)	Deployed 6 years prior to CommDash, may be displayed in Dashboard (see FCTL)	Templates feature (added as part of HOT development) used mostly by OC, PRO
FCT Log (FCTL)	Special, shared CoLT log for pushing/pulling snapshots of position-specifc CoLT log entries may be of interest to the whole team. Dashboard can display FCTL separately from or merged with individual log	PAYCOM, PRO, Marshall GC • Favorite CommDash tool across FCT, most widely used and accepted for day-to-day ops. • Has substantially increased coordination and saved time finding info by providing a single-source running real-time and historical location for console reports.
PD Status	PD Team and Voice-Video-Data readiness for POIC-managed activities – Now +/- 8 hours	• Cadre likes concept; because many inputs are manual, it's taking time for folks to build input habits; trust in what's displayed is growing
To Do List	Define/assign task assignments by position; store/show comments, completion status	TCO (biggest user by far), APOM, DMC, POD TCO uses • Timeline Reviews w/cadre, no comments via voice loop • Internal planning team comm with prep rooms (ops and prep rooms).
IMReady	Poll FCT on simple questions/statuses	• Used mostly for console readiness, special handovers, or big events. Useful for shift handovers.
Text Chat	Position-specific channels, custom channels and on-the-fly discussions, notifications	• Reduces voice traffic

V. CommDash and Human Factors Engineering - Mutual Influences

When CommDash and other HOT tool development began, users and developers agreed on Human Factors Engineering's (HF or HFE) value in display design. Alas, MOL did not have sizable resources in this area. We were able to obtain part time services from a member of the HF team in MSFC's Engineering Directorate, and one POIC cadre member had HF qualifications which he exercised when time permitted.

HF participated in ConOps revisions to ensure that as many initial considerations as possible could be modeled through use cases and prototypes, and reviewed design sketches/proposals. On previous developments, end users typically identified User Interface (UI) issues during acceptance reviews, training simulations, or flight ops, i.e., very late in the game. Thanks to our HF professionals, UI issues were resolved (or at least mitigated) in the first stages of development, and end users have become more aware of the sort of things one should look for in a UI. HF was also invited to User Evaluations, attended when available, and provided comments and feedback with visual aids. As the saying goes, one picture is worth a thousand words!

The Development Team accommodated a large number of HF inputs into the CommDash suite in response to these concerns and opportunities:

- Prevent or reduce visual and mental fatigue
- Simplify Graphical User Interfaces (GUI's)
- Minimize number of steps or clicks to access needed information
- Create more direct avenues of obtaining information, e.g., mouse hovering, visual cues such as highlighting or bolding, notifications to prevent "hunting" for information, tabular organization of information
- Customize the dashboard application per user preference
- Minimize application displays when appropriate to prevent clutter, maximize when appropriate
- Enable some applications as stand-alone tools outside of the dashboard, such as the Payload Developer Status (PD Status) application.
- Organize displayed information per HF recommended guidelines to improve efficiency and understanding

Thus far, the return on our modest investment in HF has been excellent. Suggested future goals include having dedicated HF resources within PMOD, evolving a unified interface standard based to provide consistent UIs across POIC and PMOD tools, and perhaps striving for consistency - or at least cognizance - across NASA and partner organizations. One investment that might pay off handsomely is to provide introductory HF fundamentals education and training across a broad segment of PMOD. Just a little awareness enables people to detect challenges and opportunities, then have a cogent conversation with an expert who can help resolve or capitalize on the situation.

Fig. 10 shows how CommDash relates to HF domains and topic areas relevant to flight control facilities.

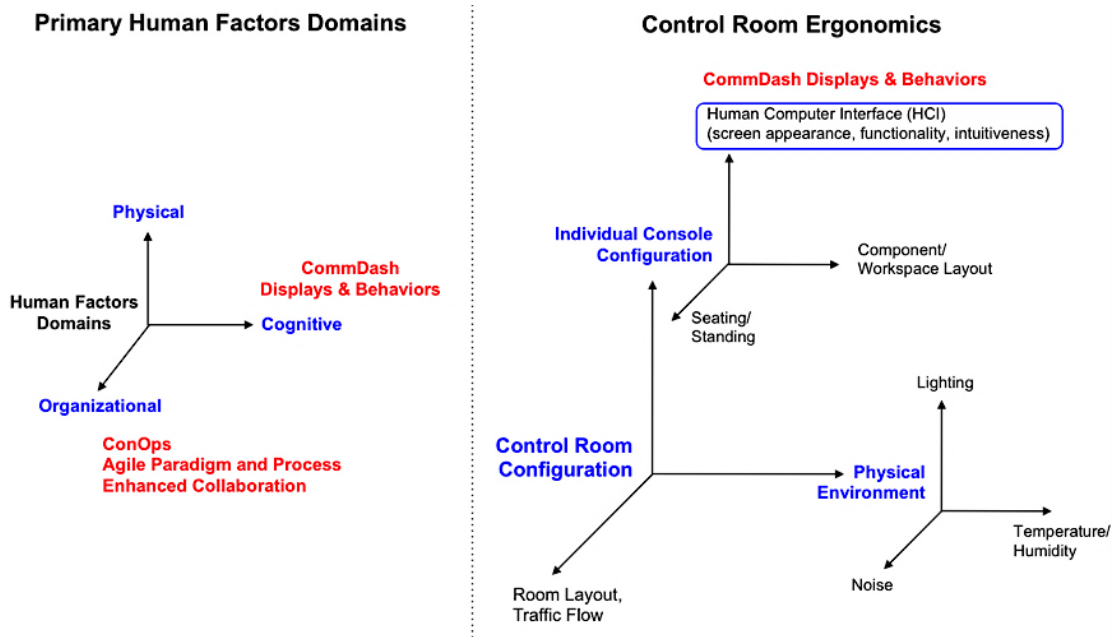


Fig. 10 Human Factors Engineering Context of CommDash

VII. Food For Thought

A. Operations

Voice loops and console logs have been primary control room tools since the beginning of space flight. Logs trace their origins to ship's records from the days of sail, and requirements for POIC cadre still invoke the notion of a "brief narrative of significant events." Early console logs were kept on paper, often used position-specific shorthand, and couldn't help but be brief due to the speed of a human hand (or lack thereof). Instant-playback recorders were generally not available. If we missed something, we made a judgment call on whether it was important enough to take up someone else's time with a "say again" query. Paper logs were used well into the 1990s, and some Space Shuttle controllers stayed with them for another ten years.

As logs became electronic, we began logging more details than previously, often filling in after the fact during quiet moments on shift or by staying afterwards. At times, the extra details and sheer number of entries obscured the "significant events" items. In our professionalism and thoroughness, we would sometimes include far more detail than necessary, and efforts to capture that level of detail posed a distraction from more important mental work, adding to overload and fatigue. Voice loop traffic of interest to multiple cadre positions was logged independently by each position, and differing perceptions might not be recognized until a disconnect became apparent some time later.

A key motivation for developing CommDash was reducing clutter on the voice loops by providing alternate venues to handle appropriate traffic non-verbally. The tracking, archiving, and everyone-sees-the-same-thing-for-items-of-common-interest visualization provided by CommDash tools is self-organizing and prevents a lot of redundant and/or divergent logging. The cadre can breathe again because content is close at hand without being "in your face" or buried. The sort of design that enables this owes much to the room for experiential discovery inherent in the development process.

B. Development

The hybridized Agile Software Development (ASD) process used to build CommDash and other POIC HOT applications enables flexible, creative development activity while respecting constraints of flight schedule-based delivery requirements, project financial planning inertia, and the schedule and task demands on POI personnel. Having developers and users working side-by-side early on identifies and mitigates technical issues while they are relatively inexpensive to fix, significantly reduces problems during IV&V and acceptance, and promotes mutual understanding of limitations and strengths that reaps benefits far beyond the initial project. It's also fun, which is genuinely healthy in the high-performance, high-stakes arena of space flight operations support.

While having developers produce a cost estimate as the first step in a project works well for situations with a stable scope, ASD projects typically involve significant unknowns at project start, which exacerbates the perils of solidifying a design approach prematurely. On very complex and abstract projects lacking well-defined concepts, there can be value in dedicating resources to a pre-project Sprint Zero in order to explore the problem and solution space (via *inexpensive,* simple, or even crude prototypes and/or simulations/exercises to discover and answer high-value questions. It is then possible to produce more viable and appropriate cost estimates, backlog definitions, and Sprint definitions for the remainder of the project. When navigating uncertainties, it is often wise to look before leaping.

References

[1] Scott, D. W., "Using Web 2.0 (and Beyond?) in Space Flight Operations Control Centers", *SpaceOps 2010*, 210896, http://ntrs.nasa.gov/archive/nasa/casi.ntrs.nasa.gov/20100021135_2010020435.pdf

[2] Scott, D. W., "Simplify ISS Flight Control Communications and Log Keeping via Social Tools and Techniques", *SpaceOps 2012*, , https://ntrs.nasa.gov/archive/nasa/casi.ntrs.nasa.gov/20120015222.pdf

[3] Cowart, H. S. and Scott, D. W., "Simplifying Operations Communication Through Application of Social Concepts", *SpaceOps 2014,* https://ntrs.nasa.gov/archive/nasa/casi.ntrs.nasa.gov/20120015222.pdf

[4] Scott, D. W., "Using Social Media in Engineering Suppor and Space Flight Operations Control", 2011 IEEE Aerospace Conference, https://ntrs.nasa.gov/archive/nasa/casi.ntrs.nasa.gov/20100017163.pdf

[5] Scott, D. W., "Communications Dashboard (Control Rooms, Take a Cue from Facebook®!) Chapter 1", 2013 IEEE Aerospace Conference, , https://ntrs.nasa.gov/archive/nasa/casi.ntrs.nasa.gov/20130013035.pdf

[6] From http://blog.adapty.com/best-practices-in-agile-development-in-ecommerce-implementation/