AVCATT-A: A CASE STUDY OF A SUCCESSFUL COLLABORATIVE DEVELOPMENT PROJECT

Don Procuniar, L3 Communications Link Simulation & Training, Arlington, Texas
Paul E. McMahon, PEM Systems, Binghamton, New York
Dennis Rushing, Simulation, Training and Instrumentation Command, Orlando, Florida

ABSTRACT

With recent advances in collaborative technology and tools, many organizations are today taking advantage of distributed development to overcome typical project management obstacles, such as compressed schedules, skilled personnel shortages, and other resource constraints. This paper is not about traditional subcontract relationships. It is about contemporary development challenges and the collaborative solutions successfully implemented by L3 Communications Link Simulation and Training on the US Army's Aviation Combined Arms Tactical Trainer—Aviation Reconfigurable Manned Simulator (AVCATT-A).

Collaborative development, as the term is used in this paper, implies the use of multiple physically separated developer and customer groups, operating as a single integrated team utilizing common processes, tools, support services, and a common technical vision all driven through a single streamlined management chain. A few years ago such a development concept might have seemed inconceivable. Today, through the use of the World Wide Web, a private company intranet, e-mail, tele- and videoconferencing, and key collaborative tools (i.e. Netmeeting, ClearQuest, ClearCase…), the three (3) primary AVCATT-A development sites (Arlington, Texas, Orlando, Florida, and Binghamton, New York) are collaboratively developing the AVCATT-A solution with their customer, the Simulation, Training, and Instrumentation Command (STRICOM), as an integral team member.

Specifically, this paper discusses the important relationships among the AVCATT-A technical architecture, the management of remote site tasking and customer involvement during the development. Techniques employed to define a project common architecture, address on-going architecture-related issues, and communicate architecture-related decisions to the full team are described. The complex relationships among build planning, project processes and tools, and the technical infrastructure are discussed, along with factors that led to the AVCATT-A specific solution.

The paper addresses the critical aspects of leadership, conflict management and site-specific culture in a collaborative environment, along with how these issues affected the AVCATT-A solution. Project communication rules and the degree of "process freedom" allowed at individual sites are discussed along with rationale. Factors driving the selection of software tools and platforms are also identified, along with lessons learned associated with the development of a common AVCATT-A workflow process.

References to other published collaborative development works are provided as an aid to the reader in comprehending the challenges being faced today on many collaborative efforts, along with practical and affordable techniques found successful on AVCATT-A.

Author Biographies

Don Procuniar is the Software Lead Engineer for L-3 Communication, Link Simulation & Training AVCATT-A Aviation Reconfigurable Manned Simulator Program. He has been employed at Link since 1980 holding numerous Software, Systems, Integration and Program Engineering positions over that twenty-year period. Donald has vast experience in large scale complex man-in-loop real time simulations including NASA Space Shuttle Mission Simulator, numerous F-16 training platforms, B-52 WST, C-130 ATS and AVCATT-A.

Paul E. McMahon, Principal, PEM Systems, provides technical and management leadership services to large and small engineering organizations. He has taught Software Engineering as an adjunct at Binghamton University, published over fifteen articles, and a book on collaborative development entitled, "Virtual Project Management: Software Solutions for Today and the Future."

Dennis Rushing is a senior software engineer at the U.S. Army Simulation, Training, and Instrumentation Command (STRICOM). He currently supports the development of the AVCATT-A and other virtual simulation systems.

INTRODUCTION

This paper is about the challenges being faced today when multiple companies and/or multiple physically separated sites join forces on an advanced technology software intensive project. It is important to note that this paper is not about traditional subcontractor relationships, but rather it is about the contemporary development challenges faced when operating as a single integrated, but physically separated, team. Many of the issues faced involve the effective utilization of processes, tools, support services, and people to aid the development and communication of a common technical vision all driven through a single streamlined management chain.

The paper provides the motivation for distributed collaborative development, and identifies four (4) misleading perceptions of why many collaborative ventures fail today. Six (6) patterns of successful collaborative operations based on previously published work are discussed. Examples of difficulties encountered when alternate patterns are utilized are included. Through the discussion insights into the challenges faced by contractors and customers when operating in a distributed fashion are provided. In this paper the words distributed and virtual are used synonymously.

MOTIVATION FOR DISTRIBUTED COLLABORATIVE OPERATIONS

Why should one care about distributed collaborative development? Simply put, success ultimately depends on the effective utilization of resources. Today the software industry is facing a critical shortage of key resources. At the same time customers are looking for solutions to new challenges based on existing products. This, in turn, is creating an increasing demand for personnel with specific product knowledge and experience. As an example from the modeling and simulation domain, because of the extensive reuse of Semi-Automated Force (SAF) products, such as OneSAF Testbed (OTBSAF), personnel with specific knowledge and experience in this technology are in considerable demand. Frequently, several groups with different skills are needed to incorporate multiple products into a single integrated solution. Often these groups exist at different geographic locations and could even be employed by different organizations.

Imagine if your organization just won a contract and the people with the skills you need aren't employed in the city where you plan to manage the development? In the past the solution would have been to "move the people". But today, given aggressive development schedules, and limited resources for retraining or relocating people, this paradigm is shifting to the use of distributed operations to "move work, not people." While the motivation is clear, many companies are today experiencing difficulties implementing collaborative operations.

MISLEADING PERCEPTIONS ABOUT COLLABORATIVE OPERATIONS

Research conducted a few years back indicates that many collaborative ventures fail due to four (4) reasons; cultural incompatibility, leadership struggles, lack of trust, and inbred notions of competition [1]. More recent research [2], however, indicates that these perceived reasons for failure are, in actuality, symptoms and that underlying these symptoms is a set of fundamental causes and remedies. A solid understanding of these underlying causes and remedies can significantly increase a distributed project's chances for success. This same research also indicates that there are a number of useful and identifiable patterns associated with successful collaborative operations. Included throughout this paper are real examples of the identified patterns as implemented by L3 Communications Link Simulation and Training and their customer, the Simulation Training, and Instrumentation Command (STRICOM), on the US Army's Aviation Combined Arms Tactical Trainer—Aviation Reconfigurable Manned Simulator (AVCATT-A) project. A brief overview of the AVCATT-A project is provided.

THE AVCATT-A PROJECT

The AVCATT-A is a networked system of systems providing combined arms training through six (6) reconfigurable manned helicopter simulators interoperating in a simulated battlefield environment. The system is housed in two trailers, and includes both a Battle Master Control and After Action Review Station. The six Manned Modules simulate pilot/copilot/gunner positions for UH-60A/L Blackhawk, CH-47D Chinook, OH-58D Kiowa Warrior, AH-64A Apache, AH-64D Apache Longbow, and the RAH-66 Comanche.

Subsequent to contract award in October 1999, project start-up activities were conducted collocated in Arlington, Texas through March 2000. In the March 2000 timeframe, an initial work split across the three primary AVACTT-A development sites was established. The Orlando, Florida site was assigned the responsibility for the Semi-Automated Forces (SAF)

Simulation, Interoperability, and Networking. The Binghamton, New York site was assigned responsibility for the User Interface, After Action Review, Mission Planning, and Mobile Facility development. The Arlington, Texas site was assigned responsibility for overall management, all Manned Modules, all hardware development, and final integration of all components.

PATTERNS OF SUCCESS

1: Collocate Key Activities Early For Common Collaborative Vision [2]

Many of us who have grown up in collocated environments tend to take for granted much of our previous common experiences shared with our teammates, as well as the ease with which we can initiate an informal discussion. For example, during start-up activities on a traditional collocated project we often take for granted the simple fact that, at any moment when a question may pop into our head, we can get up and walk down the hall and initiate a discussion with a teammate. Frequently, it is during these unplanned sessions when shared common vision begins to grow. Distributed development can diminish this advantage of proximity.

Assume the following about two engineers:

- They are physically separated
- They do not share common experiences
- They do not communicate on a regular basis

If given identical design problems to solve, what are the chances of getting the best ideas from both engineers in a single integrated solution? This is a key difference between traditional collocated and distributed projects, and it is one that is often not well understood.

For those who understand the creative process, it should not be a surprise to find out that many collaborative ventures that start out on day 1 with parallel multiple site activity end up with multiple divergent design paths leading to intense leadership struggles. It is for this reason that our first key pattern of success is to collocate key activities early to establish a common collaborative vision.

Distributed projects face the added challenge of teammates with differing backgrounds and experiences. It is therefore critical that, in addition to early collocation of key activities, project leaders be open to alternatives put forth by teammates who may have very different backgrounds and experiences from their own.

AVCATT-A Early Collocation in Arlington, Texas

The AVCATT-A project was not conceived as a traditional development project, but rather was to be based on extensive reuse. This implied that a new vision of how the project would execute was needed. As plans were established early, key leaders were involved in each step. It was not uncommon to face heated conflict at this stage of the project.

During this early phase the project architecture team consisted of over twenty (20) people including the Software Lead, System Engineering Representatives, the Process Engineer, System and Subsystem Lead Engineers, and Customer Representatives. Architecture meetings often spanned a wide range of topics including requirements, hardware, software, terminology, and process. It was not uncommon for these meetings to become "brainstorming sessions" frequently running over two hours in length. Although significant and wide-ranging architectural issues were being addressed, at times these sessions appeared to be disorganized and out of control. Even seemingly simple architectural issues took significant time to resolve because of the size and diversity of the group.

During the early collocated months of the project, the Architecture Team leaders were not quick to dictate the next evolution of the architecture. The leaders intentionally allowed the brainstorming and the apparent "out of control" activities to occur. At this stage the project leaders wanted to let all voices be heard, and all issues be raised.

Specific topics addressed by the large Architecture Team during this phase of the project included reviews of system level architecture diagrams, High Level Architecture (HLA) and Joint Technical Architecture-Army (JTA-A) compliance, operating systems, compilers and platforms, status of prototyping activities, reports on product assessments, multi-language issues, workflow issues, process guidance, and team interactions.

Many of the issues discussed related to potential reusable product constraints. Examples of products being considered for reuse on the project at this time included the Distributed Environment Manager (DEM) being reused from the L3 Link F16 project, and the Semi-Automated Force (SAF) product being reused from the OneSAF Testbed project.

In reality, the Architecture sessions, as well as other engineering activities, that occurred collocated over the first 5 months of the project were not "out of control." This was a critical time when brainstorming allowed leaders to create a vision of how the project would

evolve. Those early meetings supported the needs of the creative process, and they initiated the building of a virtual team shared experience and trust.

Customer participation in these early sessions was also valuable. Not only was the customer able to provide insight into system requirements from a user's perspective, but also a level of trust was established between the developer and the customer. The developer began to trust that the customer would not have unrealistic demands or expectations; likewise the customer developed a sense of trust that the Architecture Team was making sound decisions that would result in a product that satisfied the user's needs. Open communication and unconstrained information sharing fostered teamwork and strengthened team synergy.

Performing the initial architecture activities and creating the organizational vision at a single collocated site was a critical factor in determining the eventual success of the AVCATT-A program.

2: Establish Process, Procedures, Tools, Rules and Terminology Early [2]

Today, virtual communication is in its infancy. We are just now starting to comprehend the implications of first generation lessons on using the World Wide Web, tele- and videoconferencing, and tools such as Netmeeting, and E-mail. Beneath the obvious changes these new communication vehicles bring, lays less obvious effects on the processes, procedures, tools, and terminology required for effective collaborative operations.

In the past, many collaborative ventures have struggled when trying to establish the right level of process/procedure commonality across a virtual project. When process commonality is driven too low, site-specific culture clashes often lead to intense conflicts and leadership struggles. But when left too high, ambiguous terminology, divergent tools and inconsistent work instructions can cause a dramatic rise in the integration risk.

Many of today's distributed projects are moving toward incremental "build" approaches. A build can be considered a set of hardware and software that meets a subset of the functionality of the final deliverable product. Incremental build approaches are becoming more popular, particularly on distributed projects, because they reduce integration risk by surfacing miscommunication early. Incremental build approaches have shown that there exists a critical need for key project personnel to work together closely early.

The Build Manager is traditionally responsible for defining the functional capabilities to be included in each build. The Load Build/Configuration Management (CM) Specialist defines the process for the software to get into the load. This is usually tool specific and relates closely to engineering workflow. The Process Engineer is responsible for defining the reviews that must occur, the checklists that must be met, and any other key toll gates that must be passed, such as coding standards, test requirements, and documentation requirements. The technical architect ensures that the products moving through the process meet the Architecture Criteria.

Often, in the past on traditional collocated projects, architecture criteria, and load build procedures have been viewed as project-specific and therefore below the organizational level "process freedom line." But on collaborative ventures these issues are critical and must be discussed and agreed to early across all organizations and/or physical sites. Failure to do so has been a common contributing cause of past collaborative failures.

Experience indicates that on distributed projects the Build Manager, the Load Build/CM Specialist, the Process Engineer, and the Technical Architect(s) must work closely on a daily basis preceding the initial integration phase. This level of close interaction is usually not necessary on traditional collocated projects because personnel can rely on previously existing collocated cultures, common vision, and consistent terminology to answer many of the related questions that arise.

In many existing mature collocated environments the team effectively relies on the unwritten subcultures within the specific collocated organization. But on large distributed projects with multiple organizations, especially those that may have never previously collaborated, fundamental issues such as platform, compiler, process requirements, process toll-gates, and even basic terminology can all become major points of contention at critical points in the schedule.

AVCATT-A Common Collaborative Process

On AVCATT-A a single Software Development Plan was employed to communicate common processes, work instructions and project-specific process notes (procedures) across the project. The process notes were written at a tool-specific level and therefore needed to be consistent with the architecture criteria (discussed below) that was agreed to by all sites.

In support of reuse, the AVCATT-A process encourages the reuse of not only code and documentation products, but also associated styles and methods. All legacy code on the project is maintained in its legacy style. Legacy designs are updated employing a consistent design methodology unless the percentage of change exceeds a pre-established 30% threshold. For example, the chosen SAF product on AVCATT-A was developed to a specific design and coding standard. Following the AVCATT-A reuse-focused process, modifications to the SAF product were made without deviations from the original design methodology and coding style.

Support for varying methodologies could have added significant integration risk to the AVCATT-A project. However, this risk has been managed on AVCATT-A through a number of vehicles, including well-defined Architecture Compliance Criteria.

The Architecture Compliance Criteria on AVCATT-A was developed to judiciously specify hardware platform, compiler, and interfacing requirements. Ensuring all products approved on AVCATT-A meet the criteria is key to managing the integration risk. This approach allows the project to leverage existing strengths of not only existing products, but also the processes and people that support those products. This notion has been referred to as a "Freedom Line" in other published works on distributed development [2].

As the vision on AVCATT-A for the common reuse-focused collaborative procedures became clear, formal presentations were provided to the full Engineering team across all sites to communicate and cultivate a sense of ownership throughout the virtual organization. Because the procedures were developed with collaboration in mind, and because they took into consideration the specific needs of each site, they were accepted across the full project.

It is important to note that AVCATT-A was able to achieve greater process/procedure/tool commonality across sites than many other distributed projects. Unique project characteristics must be considered in determining the right level of process/procedure/tool commonality for other projects. The deeper we can drive commonality (beyond process, and into procedures, tools, and terminology), the more we can reduce the chances of miscommunication.

AVCATT-A Multi-Level Integration Plan

A key to the AVCATT-A success was the development of a multi-level detailed integration plan. This included integration planning for all work activities at all sites, and for incremental integration of the final product. The final integration for each incremental build occurs in Arlington, Texas.

On AVCATT-A through Build 1, no personnel from remote sites were required to travel to Arlington, Texas to perform integration & test. Key to this success was the close interaction among the Build Manager, the Configuration Management/Load Build Specialist, the Process Engineer, and technical architects.

AVCATT-A Common Process Notes & Tools

On AVCATT-A the Rational ClearQuest/ClearCase product-set was the chosen Work-Flow/Configuration Management (CM) tool for all sites. A complete set of workflow states were defined and captured in the ClearQuest tool. These states, of course, needed to be consistent with the detailed process notes that had been developed on the project by the process engineer.

It is also worth noting that at proposal time multiple compilers (Green Hills, GNAT) and multiple Operating Systems (VxWorks, Linux, and Windows) had been proposed to support the extensive planned reuse of legacy software. Subsequent to contract award, this approach was reviewed and modified driving the majority of the real-time software to the GNAT family of compilers and the LINUX operating system. This was done to reduce life cycle maintenance costs, reduce training cost, and provide common development tools across all sites. In this case, the Architecture Team looked beyond any one group's view in the interest of the overall project.

On many collaborative efforts intense leadership struggles result when tools and hardware platforms are not agreed to early. On AVCATT-A key decisions made early, documented, and communicated to the full team with respect to common process, procedures, tools, compilers, and operating systems all contributed to minimizing cross-site leadership struggles during the critical integration phase. Failure to make key decisions early and communicate the results to the full team is a common pitfall witnessed on many past unsuccessful distributed efforts.

Communication & Rules [2]

One significant factor that contributes to the success or failure of a distributed project is the ability of team members to communicate openly and efficiently. On many collocated projects, team members often have established relationships based on previous experience and they understand what is expected of them as a team member based on local culture. Nevertheless, even when virtual team members have worked together in the past, the physical separation and differing organizational reporting structures can sometimes cause

problems. It is for this reason that on a virtual project more rules of communication need to be developed and documented.

AVCATT-A Common Rules

The "Gang of Four", discussed at greater length later in this paper, was a small core architecture group constructed to communicate architectural decisions more rapidly. As an example of the need for communication rules, the "Gang of Four" developed a position paper on a topic referred to as "Isolated Operation." Unfortunately, before all team members reached consensus, the paper was formally released as project direction. When a team member who had been away returned and reviewed the paper, he disagreed with the approach taken. Eventually, the position paper had to be rescinded, revised and re-released.

After this incident, the group instituted the rule that if any member of the group was missing, the group could still meet, but that no position papers could be released without the full consensus of all four members. Although, this may seem trivial and obvious, it is the absence of these types of communication standards on a distributed project that can lead to discord and ultimately influence the successful completion of the project.

From one perspective, this example is not unique to collaborative projects. The importance of keeping "in process" information inside the team until all team members agree, is certainly relevant to any team, collocated or distributed. But with virtual collaborative teams it is often increasingly difficult to keep "in process" information inside the team. This is because we often find greater pressure on virtual project personnel to report internal team information at inappropriate times through inappropriate channels. For more information on this subject refer to Chapter 5 of the referenced text. [2]

Terminology [2]

Often when personnel with divergent backgrounds and experiences are pulled together on a collaborative effort, terminology can be a significant hindrance to effective team communication. One way to resolve this is through working groups that brainstorm, document, and communicate agreed to terminology.

AVCATT-A Terminology

On AVCATT-A, two areas were identified where terminology was crucial and small groups were convened to resolve related issues:

- Architecture
- WorkFlow-Build Process

Each is discussed briefly below.

Often, the AVCATT-A "Gang of Four" dealt with clarification of terminology. Examples of terms that required clarification are Restart, Reconstitute, and Reset. While such terms may be common within the Modeling and Simulation Domain, each can have subtle differences in interpretation based on the differing past experiences of personnel. Because of the differing past experiences on collaborative efforts it becomes increasingly important that we define precisely and communicate terms across all project sites, including the customer. On AVCATT-A the documentation of many key terms was initially captured in position papers that engineers could then utilize in system development and implementation and the completion of formal deliverables.

The small working group, discussed earlier, that included the Build Manager, the Process Engineer, the Load Build/CM Specialist, and the Technical Architect spent many hours brainstorming and clarifying terminology. Among the terms formally defined and communicated to the full team were "Sandbox" Environment, "Development" Environment, "Test" Environment, "Unit" (tailored for Reuse), "In-Work", "Resolved", "Recommended", and "Authorized". The last four are examples of ClearQuest Workflow states.

The team intentionally chose terminology that did not have a specific meaning from previous known projects. This enabled everyone to learn a common new language while minimizing miscommunication.

Because of the loss of a common culture on virtual projects it becomes essential for process, procedure, tools and terminology all to be agreed to early on the project. This effort must also define the "process freedom line" which indicates where a given site is allowed to leverage site-specific procedures and tools.

It has been recommended that virtual project lessons drive written virtual project rules (guidelines) to aid engineers. These rules should include team meeting rules, use of e-mail, and guidance in proper etiquette for tele- and video conferencing. For more information on first generation virtual communication lessons and rules see chapter 5 of the referenced book. [2]

3: Allocate Work Based On Architecture and Resource Availability [2]

When surveyed about distributed development, many managers have expressed a concern related to not knowing if a remote team member is "doing the right thing". But think, for a moment, about how a manager

gains confidence that a new team member, who has been given a design task, is in fact "doing the right thing" in a traditional collocated environment. Often, when a manager assigns a new engineer a task, a senior engineer is assigned to guide the new engineer.

Traditionally, many think of architecture as a technical issue, and work split as a separate and distinct management issue. But in practice work split decisions can fracture a sound architecture. Furthermore, a sound technical architecture can actually provide one of the best task communication and coordination techniques.

Now think again about how a senior technical mentor guides that new engineer. The most effective mentors guide by listening first, and then providing feedback that ensures approaches chosen fit within the range of an organization's acceptable solutions. In other words, effective mentors guide through the vision of a technical architecture. Although the process described is commonplace, the relationship being described among architecture, work split and tasking has not always been well understood. [2]

When used appropriately, a sound technical architecture can go a long way to addressing a manager's concerns about whether a remote team member is in fact "doing the right thing". Often it is through informal architecture-centric discussions that teammates gain the real insight needed to accurately meet task expectations within the organization. But for architecture to be effective as a task communication aid, the work split definition across physically remote sites must follow the architecture definition, not the reverse.

Too often we see work split decisions made without due consideration to the technical architecture. When work split decisions are forced prior to architecture definition, we often find distributed projects suffering from "fuzzy" task responsibilities and technical leadership struggles. Without a well-defined architecture, remote groups often find themselves heading down inconsistent paths leading to project conflict and control struggles. As an example, the choice of computer hardware platform has been a topic of intense inter-site battles on many distributed efforts. [2]

While Architecture must precede work split, this is not meant to imply that a vision of the work allocation across teaming sites and organizations should not be established early. However, it is intended to imply that organizations/sites should expect a refinement, or even a major change, of work allocation as the architecture evolves in order to ensure the most effective work allocation has been made based on the skills, resources and specific product knowledge available across the entire project. Another common pitfall often witnessed on past virtual efforts is an overly rigid work split that doesn't stay in step with an evolving architecture and with a changing resource availability picture. [2]

It is crucial to keep in mind that a prime motivator for distributed operations is the "move work, not people," paradigm discussed earlier. To gain the full benefits, this paradigm should be employed throughout the project life. This is important because the resource picture on distributed projects, like any project, can change from phase to phase and from site to site. A key to distributed project success is a work split agreement that allows for tasking refinements across remote sites consistent with the architectural evolution, and with changing project resource requirements and availability.

AVCATT-A Example of Allocation of Work based on Architecture & Resource Availability

As mentioned in the overview, the Orlando site was assigned the responsibility for the Semi-Automated Forces (SAF) Simulation, and Networking, while the Binghamton site was assigned responsibility for the User Interface, and the Arlington site was responsible for final integration of all components.

Because of the heavy reuse-focus on the project, product knowledge evolved over time. As an example, personnel assigned to the SAF spent considerable time analyzing the existing OneSAF product to fully understand its architecture. As a result, a "tight coupling" between the front-end (User Interface), and the back-end (SAF Simulation) became apparent. This meant that changes to either would require team members assigned to the front and back ends to interact frequently. Studies have shown that when team members must interact frequently and for short intervals that remote operations can be a significant deterrent to success. [3]

As a result, based on the added knowledge gained on the SAF product architecture, a refinement of work split was made allocating the front end user interface SAF work to the Orlando site. In essence, evolution of the system architecture dictated a refinement of project work split and task allocation.

Another example of work split refinement can be seen in the Distributed Environment Manager (DEM) task. The DEM is a Distributed Interactive Simulation (DIS) component that provides the software that manages interfaces between AVCATT-A components. This task was originally allocated to Arlington, Texas.

During early development activities it was identified that the BMC and AAR stations would also require a DEM-like function. Orlando was already tasked with providing a SAF DIS interface and with providing gateways for external system interfaces. As a result of numerous architectural meetings it was strategically decided to move the DEM development to Orlando in support of interface management design commonality.

The decisions to allocate the complete SAF/DEM efforts to the Orlando site were based on a number of factors beyond the technical architecture. For example, SAF and DIS interface expertise was known to exist in the Orlando area. In addition, by managing interfaces at a single site, it becomes easier to refine work allocations across sites, provides tighter control of interfaces and cross-checks design implementations.

AVCATT-A's approach was, and continues to be, to leverage the right assets/skills for each task, regardless of where those skills may physically be located. This strategy is consistent with the "move work, not people" paradigm.

4: Establish a Virtual Culture, Collaboration and Information Sharing Methodologies and Tools [2]

A "Virtual Culture"[2] is a simple, yet powerful concept that brings an information-age perspective to the notion of culture. You can think of a "Virtual Culture" as the infrastructure that supports effective communication across distributed project sites.

The Virtual Culture, unlike traditional collocated engineering cultures, is product-oriented. It is not intended to replace past traditional collocated cultures. In fact, it is not recommended that you try to replace strong local cultures. Rather, past experience indicates that teammate strengths should be leveraged within their proven environments. Virtual Cultures can be implemented through a Web Site, or through a Shared Directory System, together with collaboration tools, such as Netmeeting, e-mail, and tele- and videoconferencing equipment. Refer to the referenced text [2] for more information on Virtual Cultures.

It is worth noting that a key difference between a virtual culture, and a traditional culture is found in its formality. Experience indicates that an effective virtual culture cannot be as informal as a traditional culture. In other words, more needs to be "WRITTEN DOWN". [2]

AVCATT-A Virtual Culture Implementation

As an example, the AVCATT-A project implemented a Virtual Culture through a combination of shared directories on a private intranet, a project Web Site, and an Internet accessible server to pass large data files to the customer. Through these communication vehicles, the System Engineering Management Plan (SEMP), the Software Development Plan (SDP), Engineering Processes and Procedures, Architecture Position Papers, Meeting Minutes, Peer Review Results, Software and System Development Folders, and Configuration Managed Artifacts are all available to team members, regardless of physical location.

One lesson learned related to the size of the project specific Software Development Plan (SDP) and related project-specific process notes. The original project vision called for an SDP of no more than 20 pages, and no more than 20 process notes, each 5 pages or less. The SDP ended up three times as large as planned, and the project developed 35 project-specific process notes, many of which were over 10 pages in length. The project continues to streamline and evolve its plans and process notes, but upon review there is good reason for the increase over the planned size. As mentioned previously, on a distributed project we can no longer rely on collocated cultures to convey project-specific tool, terminology and work instruction information that has been agreed upon to be common across all project sites. Therefore, to reduce risk of miscommunication, anticipate an increase in the written word in areas that are common across sites and organizations.

It should be emphasized here that a return to the days of voluminous milestone-type documentation is not being recommended. Rather, the emphasis on the written word is specifically focused on areas that must be communicated across multiple sites/organizations. [2]

The use of tele and video-conferencing along with collaboration tools such as Netmeeting, the immediate access to both controlled and in process artifacts, and the use of common terminology and tools all work to support a single integrated common process and culture on AVCATT-A that is not limited by physical location. On AVCATT-A individuals work as teammates building strong bonds and relationships often with others they have never met face-to-face. When it works, like it does on AVCATT-A, physical location truly becomes a non-issue.

5: Establish an Organizational Structure Conducive to Successful Collaboration [2]

Experience indicates that at the top end of the organization where a breath of issues must be

addressed, the Integrated Product Team (IPT) structure tends to function well. This is the level where "heads-up" activities exist. By "heads-up" activities we mean work that must look across the multiple sites and organizations of the project. However, it has been found that a strict IPT structure is weak when it comes to producing products that include detailed design, code, and test cases. It has also been found to have weaknesses when it comes to the implementation of common architectural solutions across a distributed project. [2]

Where the "real" engineering, or what we refer to as "heads-down" work, occurs, we have found that on distributed efforts a "hybrid" of IPTs and functional groups is often more effective. When we use the term "heads-down" work we mean the engineering work associated with building and testing actual products.

An example of why we recommend this structure can be seen in the need for an infrastructure implementation group that provides "common services" that multiple product development teams may need across remote sites. Too often, when virtual projects try to drive a strict IPT structure deep into the organization, responsibility for critical common services is lost. This is because when a strict IPT structure is employed at lower levels of the organization, we often find that each of those lower level IPTs focuses almost exclusively on their own specific product. As a result, each solves their own "specific" problem in their own "specific" way, and therefore commonality is lost. [2]

AVCATT-A Successful Collaborative Organizational Structure

The original AVCATT-A organization included a Systems Engineering Integration and Test (SEIT) IPT, a Manned Module IPT, and a Training Environment IPT. Early in the project this organization was found to be insufficient in establishing an integrated architectural solution.

Subsequent to the first Engineering Design Review (EDR), a new single streamlined Engineering Management team was put in place to provide engineering guidance and oversight. This team consisted of a single Engineering Manager responsible for all engineering activities, a Software Manager responsible for all Software and the Technical Architecture, a Hardware Manager responsible for all hardware development, a Systems Manager responsible for all program requirements and final government testing, and two functional managers responsible for the Training Environment and Manned Module IPTs.

Once the streamlined management team was in place, the AVCATT-A Engineering Manager held weekly Engineering reviews addressing development efforts at all sites. Monthly cost reviews were also held with all AVCATT-A Cost Account Managers to cross check information being provided during Engineering Reviews. Periodic Senior Management reviews were held to provide status to executive management on risk and problem areas.

The streamlined engineering management chain provided a single decision point for resolution of Architecture issues (Software Manager), and multi-site Engineering issues (Engineering Manager, or Senior Management). The lack of a single point for conflict resolution has been a major contributing factor to many past distributed project failures [2].

By placing the Architecture Team directly under the Software Manager who was responsible, and accountable, for all software across all sites, common architectural decisions could more effectively be deployed into the product implementations. Recall that the Software Manager was empowered to move work as necessary among the sites. This organizational structure allowed architecture activities to truly be integrated directly into the implementation activities. Too often, on past distributed projects, we have witnessed ineffective Architecture Groups that effectively "sit off to the side" making decisions which "have no teeth", and therefore fail to get implemented.

AVCATT-A EDRs & Architecture Team

The original project plan was to hold several large EDR events during the life of the program in support of stakeholder review and approval. Although effective, there was a large overhead associated with these reviews. In addition to the considerable amount of travel required accommodating government, user, engineering, test, logistics and management representatives, there was a significant effort expended by the developer in preparing for and participating in these reviews.

Also, given the project's tight schedule, and difficulties in reaching consensus on technical issues with large groups, in the summer of 2000 it was recognized that a change in the way the EDRs were being conducted and in the way the architecture was being evolved and communicated was needed. Recall from our earlier discussion that the early project Architecture Team (AT) consisted of over twenty (20) people. This led the customer and L-3 to redefine the process for EDRs and to create the concept that became known as the AVCATT-A "Gang of Four."

The "Gang of 4" Concept

The "Gang of Four" concept was based on the recognition that architectural decisions needed to be made and communicated more rapidly, and a small core team could accomplish that task more effectively than a large committee. This core architecture group included representatives from each of the major development sites (3), plus a computer systems specialist.

The Deputy Software Lead provided a facilitator role for the group. This was found to be necessary to keep the group focused on priority risk-based issues. When a consensus could not be reached the Software Lead was called upon to provide an arbitrator role.

In support of the "Gang of Four," a process was defined by which issues were raised to the group, handled, and results communicated. This included the production of technical "Position Papers" establishing direction on key architectural issues to the team.

The team was careful not to take on the full AVCATT-A design responsibilities, but rather provide crisp architectural guidance to engineering. In many cases, this guidance equated to a bounding or interpretation of the requirements.

Examples of issues addressed by the group included clarification of terms, operational timelines, checkpoint design, isolated operation, reset fidelity, and interoperability guidance. After agreement on an issue was reached by the "Gang of Four", a position paper would first be presented to the larger Architecture Group through a specially called session, and then to all of Engineering at regularly scheduled training sessions held on Monday mornings.

The "Gang of Four," the small working groups, and clearly tasked engineering teams across the country are all examples of the effective use of small teams on a large complex project. Reference [4] provides more information on operating a large project as a collection of small projects. We have found on AVCATT-A that the best role for leadership is one of service. Put the vision in place, and then "let the horses run."

Customer involvement also evolved in accordance with this principle. It was determined that customer participation in the process could be more efficiently accomplished through small, collaborative and less formal distributed reviews in lieu of the large, formal collocated meetings. Initially, the large formal EDRs were necessary to establish a shared vision and to ensure common understanding of requirements. These were eventually replaced with smaller design reviews with only the essential development and customer representatives participating remotely. These lower level reviews created a common understanding of how the overarching architecture was being transformed into a viable system that satisfied the specified requirements. Hardware and software reviews were organized to correspond with natural system architectural boundaries. Customer representatives participated remotely through Netmeeting and teleconferencing. Review materials and minutes are distributed and maintained on the customer web site.

6: Vigilantly and aggressively identify and resolve conflicts stemming from cultural incompatibility, leadership struggles, lack of trust, and inbred notions of competition [2]

Unlike most traditional collocated projects, virtual projects face the added challenge of teammates with differing backgrounds, experiences and technical expectations. This situation can give rise to frequent and often intense conflict. Conflict in itself is not bad. In fact, healthy conflict can invigorate a team as it reaches resolution and overcomes challenges. While most of the conflicts faced on distributed projects are not really different from those traditionally faced in a collocated environment, all too often on distributed projects these conflicts remain unresolved for long periods of time. In analyzing this situation it has been found that many organizations have effective ways to deal with site level internal conflict. Unfortunately, conflict that crosses organizational or site level boundaries is often not dealt with as effectively.

Managers must be trained to detect the warning signs of unhealthy distributed project conflict, and vigilantly and aggressively seek resolution. Furthermore, virtual project organizations must be structured in a manner that supports a strong conflict management process, in particular, for conflicts that cross organizational or site boundaries. For more information virtual project conflict see the referenced text. [2]

AVCATT-A Conflict Resolution & Management In Action

Key to the single streamlined management chain on AVCATT-A is a single decision point for conflict resolution for all Software and Architecture issues (Software Manager), and a single decision point for Engineering issues (Engineering Manager, or Senior Management). The project level conflict procedure called for conflicts to be handled at the lowest level in the organization, but any conflicts in the software or architecture areas that could not be resolved at the lower levels were raised to the Software Lead, who had

been well trained in listening first, and taking timely action. It was also critically important that the Software Lead was empowered to make decisions in the best interests of the project, regardless of site dependencies. Too often, on past distributed efforts, when the "right" decision for the project crossed organizational boundaries, the conflict resolution process was observed to break down. This has been identified as a major contributing factor to many past distributed project failures. [2]

Examples of project conflict that crossed organizational boundaries and were handled effectively were discussed earlier in the SAF and DEM work split examples. In each of these cases the Software Manager looked beyond the single site perspective, making timely decisions in the best interests of the overall project. Similarly, for systems and hardware engineering issues that crossed site boundaries, senior management involvement in setting priorities across sites and acquiring needed people resources for the project in a timely manner, helped resolve conflicts that potentially could have impeded progress.

CONCLUSION

Distributed collaboration is a viable and necessary solution to the challenges being faced in developing complex integrated products with stringent schedule and resource constraints. On AVCATT-A, a greater commonality of processes, procedures, tools and terminology was achieved than on many distributed efforts. Achieving this commonality required early, collocated establishment of a global vision, as well as the early definition of the system architecture. The common experiences and culture shared by many of the leadership team certainly contributed to the degree of commonality realized.

Process, procedures, tools, and terminology commonality provides the distinct advantage of being able to move work seamlessly across multiple sites while minimizing associated training costs. Since project personnel at all sites can be trained in a common set of tools, processes and procedures, moving work across the country is as simple as moving it into the next cubicle. The AVCATT-A Collaborative Environment supports the paradigm of "move work, not people," allowing access to a broader and deeper set of skills and resources in support of project and company goals. The allocation of work across various sites can only be successfully accomplished when consideration is given to the characteristics of a predefined system architecture, in addition to the distribution of technical resources.

It is important to note that the level of process, procedure and tool commonality achieved on AVCATT-A may not be appropriate for all collaborative ventures. The appropriate degree of commonality, as well as a suitable level of process freedom, depends on project-specific characteristics. In each case, the cost of commonality must be weighed against the integration risk.

Common vision, managed conflict, and communication focus, along with clear work split, common architecture, and common terminology, are all factors necessary to succeed in a collaborative effort. However, above all else the success of AVCATT-A must be attributed to its people. Although personnel do not all live in the same city the project has established its own "virtual culture" that provides an important framework through which its people effectively communicate.

It is critical that interpersonal bonds and trust among a distributed team be established early. It is also crucial that project leaders take an active role in instituting and encouraging relationships and fostering trust. This proved invaluable to the success of AVCATT-A

While AVCATT is a successful collaborative venture, it is not without its share of conflict. All projects, in particular all virtual projects, can expect conflict. On AVCATT-A, conflicts are used productively to focus vision, attain consensus, and improve both product and processes. Leaders are continuously on the lookout for unhealthy conflict and aggressively seek resolution and opportunities to transform potentially disruptive situations into positive experiences. The team is able to resolve conflicts and advance, because of the trust and respect shared by all team members, including management, the Architecture Team, the distributed engineering team, and the customer.

REFERENCES

1. Norton, Bob and Smith, Cathy, Understanding the Virtual Organization, 1997 Barron's Educational Series, Hauppauge, NY, pg. 68.
2. McMahon, Paul E., Virtual Project Management: Software Solutions for Today and the Future, St. Lucie Press, An Imprint of CRC Press LLC, Boca Raton, 2001.
3. Gindele, Mark E., and Rumpf, Richard, Effects of Collocating Integrated Product Teams, Program Manager, July-August 1998, p. 38
4. McMahon, Paul E., "Integrating Systems & Software Engineering: What Can Large Organizations Learn from Small Start-Ups?," Software Technology Conference, Track 2 Process Improvement, May 2001