

# **SAND REPORT**

SAND2002-2064  
Unlimited Release  
Printed July 2002

## **2002 SNL ASCI Applications Software Engineering Assessment Report**

C. Michael Williamson, Harvey C. Ogden, Kathleen A. Byle

Prepared by  
Sandia National Laboratories  
Albuquerque, New Mexico 87185 and Livermore, California 94550

Sandia is a multiprogram laboratory operated by Sandia Corporation,  
a Lockheed Martin Company, for the United States Department of  
Energy under Contract DE-AC04-94AL85000.

Approved for public release; further dissemination unlimited.



**Sandia National Laboratories**

Issued by Sandia National Laboratories, operated for the United States Department of Energy by Sandia Corporation.

**NOTICE:** This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government, nor any agency thereof, nor any of their employees, nor any of their contractors, subcontractors, or their employees, make any warranty, express or implied, or assume any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represent that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government, any agency thereof, or any of their contractors or subcontractors. The views and opinions expressed herein do not necessarily state or reflect those of the United States Government, any agency thereof, or any of their contractors.

Printed in the United States of America. This report has been reproduced directly from the best available copy.

Available to DOE and DOE contractors from  
U.S. Department of Energy  
Office of Scientific and Technical Information  
P.O. Box 62  
Oak Ridge, TN 37831

Telephone: (865)576-8401  
Facsimile: (865)576-5728  
E-Mail: [reports@adonis.osti.gov](mailto:reports@adonis.osti.gov)  
Online ordering: <http://www.doe.gov/bridge>

Available to the public from  
U.S. Department of Commerce  
National Technical Information Service  
5285 Port Royal Rd  
Springfield, VA 22161

Telephone: (800)553-6847  
Facsimile: (703)605-6900  
E-Mail: [orders@ntis.fedworld.gov](mailto:orders@ntis.fedworld.gov)  
Online order: <http://www.ntis.gov/ordering.htm>



SAND2002-2064  
Unlimited Release  
Printed July 2002

# **2002 SNL ASCI Applications Software Engineering Assessment Report**

C. Michael Williamson and Harvey C. Ogden  
Infrastructure and Information Systems Center

Kathleen A. Byle  
Nuclear Waste Management Program

Sandia National Laboratories  
P.O. Box 5800  
Albuquerque, NM 87185-1137

## **Abstract**

This document describes the 2002 SNL Accelerated Strategic Computing Initiative (ASCI) Applications Software Quality Engineering (SQE) Assessment and the assessment results. The primary purpose of the assessment was to establish the current state of software engineering practices within the SNL ASCI Applications Program.

## Table of Contents:

Executive Summary .....	5
Introduction.....	5
Assessment Conduct and Scope.....	5
Assessment Results.....	5
Assessment Recommendations.....	7
Easily Implemented Improvements .....	7
Other Opportunities for Improvement .....	8
Conclusion .....	8
Conduct and Scope of the Assessment .....	9
Schedule.....	9
Critical Success Factors .....	12
Sponsor Support.....	12
High-performance Teams .....	12
Site Coordination .....	13
Development of a Scoring Process .....	13
Roles and Responsibilities .....	14
Assessment Process & Results.....	15
Assessment Process .....	15
Assessment Team Scores and Code Team Scores.....	15
Interview Exit Questions.....	18
Better Practices .....	25
Requirements Phase (Practices 1a – 1g) .....	25
Development: Design Subphase (Practices 2a – 2f) .....	26
Development: Implementation Subphase (Practices 3a – 3d) .....	27
Development: Test Subphase (Practices 4a – 4f) .....	27
Release Phase (Practices 5a – 5e).....	28
Project Management (Practice 6a).....	28
Tracking and Oversight (Practices 7a – 7c).....	28
Risk Management (Practice 8a).....	29
Support Elements (Practices 9a - Practice 12e).....	29
Assessment Recommendations.....	30
Easily Implemented Improvements .....	30
Documenting Quality Improving Work.....	30
SQE Resources.....	31
Issue Tracking.....	31
Metrics .....	31
Other Opportunities for Improvement .....	32
Program Issues .....	32
SQE Resources.....	32
SQE Tools.....	32
Records Management.....	32
AQMC.....	33
Code Teams .....	34
References.....	36

Appendices..... 37

    Summary of Suggested Changes for the SQE Practices Document [2]..... 38

    First Time Assessment Lessons Learned..... 40

    Assessment Team Scores..... 41

    Exit Question Responses..... 43

        Positives (What is working in your environment.) ..... 43

        Barriers..... 46

        The Opportunities (Expected outputs of the assessment) ..... 49

Team Plots ..... 52

    Team 1 ..... 53

    Team 2 ..... 53

    Team 2 ..... 54

    Team 3 ..... 55

    Team 4 ..... 56

    Team 5 ..... 57

    Team 6 ..... 58

    Team 7 ..... 59

    Team 8 ..... 60

    Team 9 ..... 61

    Team 10 ..... 62

    Team 11 ..... 63

    Team 12 ..... 64

    Team 13 ..... 65

    Team 14 ..... 66

    Team 15 ..... 67

    Team 16 ..... 68

    Team 17 ..... 69

    Team 18 ..... 70

    Team 19 ..... 71

    Team 20 ..... 72

    Team 21 ..... 73

    Team 22 ..... 74

    Team 23 ..... 75

    Team 24 ..... 76

**Sandia is a multiprogram laboratory operated by Sandia Corporation, a Lockheed Martin Company, for the United States Department of Energy under contract DE-AC04-94AL85000.**

## **Executive Summary**

### ***Introduction***

This document describes the 2002 SNL Accelerated Strategic Computing Initiative (ASCI) Applications Software Quality Engineering (SQE) Assessment and the assessment results. The primary purpose of the assessment was to establish the current state of software engineering practices within the SNL ASCI Applications Program.

The assessment was conducted at the request of the ASCI Advanced Applications Program Manager, Mike McGlaun.

### ***Assessment Conduct and Scope***

The assessment followed the process described in the *ASCI Applications Sandia Procedure (AASP) 13-1* [1] and utilized the GAP assessment tool described in the *ASCI Applications Software Quality Engineering Practices* [2], which implemented practices described in the *ASCI Software Quality Engineering: Goals, Principles, and Guidelines* [3].

The assessment began on 12/11/2001 with a pilot assessment of a single code team, and continued until a total of 24 code teams had been assessed. The assessment concluded 3/12/2002 with an out-briefing for the assessment sponsor.

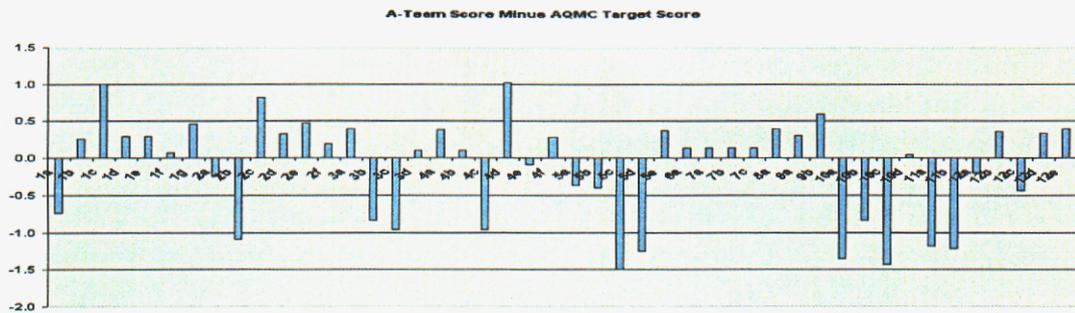
The name of the codes and their scores will not be publicly released. The scores will be associated with a code number, not the code name. Management made this commitment to foster a more open interchange between the code teams and assessors, because this was the first time many of the code teams were assessed, and because there was no time for the code teams to make any changes in their practices after the publication of the *ASCI Applications Software Quality Engineering Practices* [2].

### ***Assessment Results***

Code team results were consolidated into a single draft data set for analysis. Results and plots were generated both at the individual code team and program levels. Four types of plots are included in this document:

- Program level plots representing the results for all teams.
- Program level plots representing average results.
- Assessment specific plots. (e.g. assessment exit question results.)
- Team specific plots (provided in the appendix).

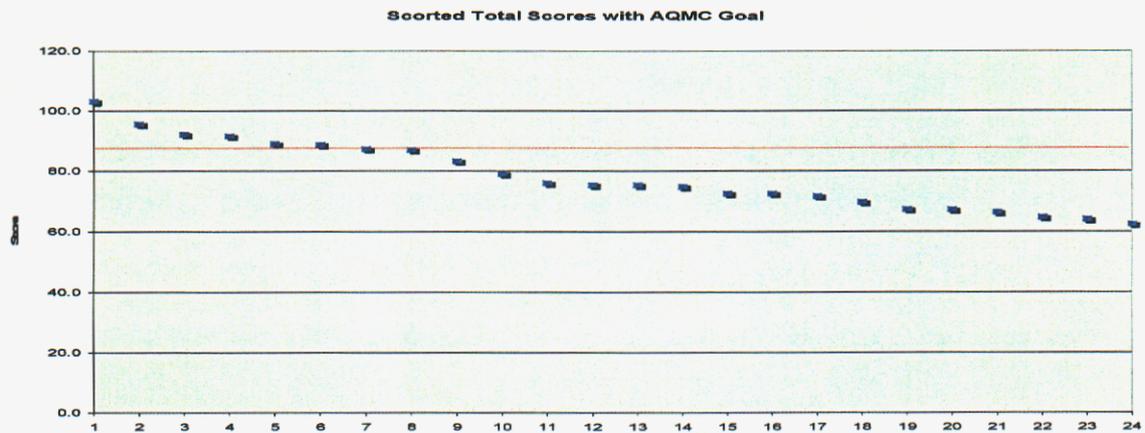
Three representative plots are provided in this section.



**Figure 1 - Average Scores Minus Target Scores (Across all 46 Practices)**

The first result of interest represents a program level gap assessment of the forty six practices [Figure 1]. A gap assessment is an method of determining the difference (or gap) between the current state and a desired state of a project or program. In figure 1, results above the center line represent practices for which the program is exceeding the AQMC targets. Results below the center line represent practices not meeting the AQMC targets. Based on the AQMC targets for the forty six practices [2], twenty eight practice targets are currently being met at a program level. A total of eighteen practice targets are not currently being met.

A second program level plot shows the scatter of total team scores for the twenty four teams [Figure 2]. The AQMC total target score (eighty seven) is indicated by the red line.



**Figure 2 – Sorted Total Scores (with AQMC goal)**

As shown in plot, eight of the code teams scored at or above the total AQMC goal.

Code team members (and code team managers) were interviewed as part of the assessment process. At the conclusion of the interview each was asked to identify those things that are working well, those things that are barriers, and their assessment

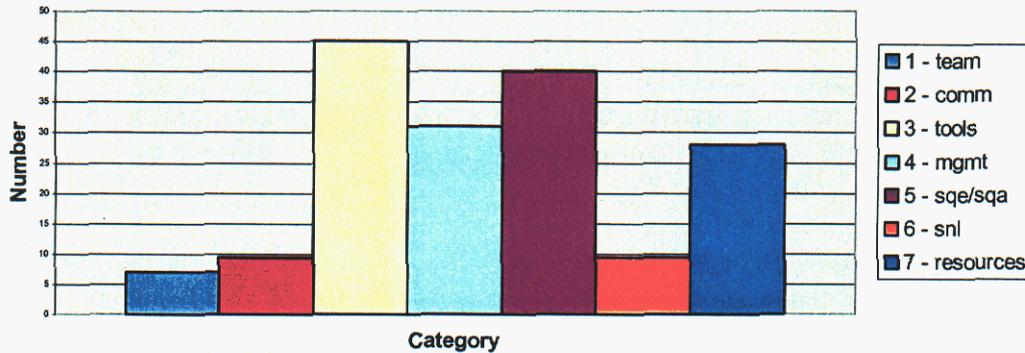


Figure 3 – Code Team Responses to “Barriers”

expectations.

The third plot [Figure 3] represents a summarization of the comments made by code team members to the second question, “what are the barriers”. The greatest number of comments related to Software Quality Engineering (SQE) Tools (or lack thereof), SQE/Software Quality Assurance (SQA) resources, and Management. The comments relating to SQE tools indicated a need for more/better SQE tools and SQE tool support. Comments on SQE/SQA resources indicated that many code teams do not have adequate access to SQE/SQA resources. Comments on management issues generally related to schedule and priorities.

### Assessment Recommendations

Recommendations fall into two main categories; “easily implemented improvements” and “other opportunities for improvement”. For a recommendation to be considered an easily implemented improvement it must be achievable with little cost or schedule impact.

#### Easily Implemented Improvements

- Document Quality Improving Work; generate and retain meeting notes, discussion notes, critical decisions, etc.
- SQE Resources; identify the existing pools of SQE knowledge at SNL and ensure that code teams know how to access these SQE knowledge pools.
- Issue Tracking; code teams need access to a web based issue tracking tool; many teams are using Bugzilla (a no-cost web based tool) at this time.
- Metrics; decide what metrics are of value to the program and code teams; start planning and collecting metric data.

## Other Opportunities for Improvement

- Issue Tracking; Issue Tracking is needed at the Program level.
- SQE Training; continue (and enhance) the SQE training that preceded the assessment.
- SQE Resources; code teams need access to SQE resources. This is both a resource and budget issue.
- SQE Tools; align, improve, and enhance SQE tools as possible. The tools and tool sets in use today are helping.
- Records management; the code teams need directions and tools in this area soon.
- *ASCI Applications SQE Practices* [2]; several enhancements/changes are recommended in the body of this document.
- Program Infrastructure; create/update program processes and procedures that are needed to support the SQE effort (e.g. AASP 13-1 [1]).
- Quality Document Issuance; it is time consuming and difficult to modify/update some documents (e.g. SAND Reports). Determine what types of documents must be issued as SAND Reports, and what types of documents might be issued using other mechanisms.
- Formation of an informal ASCI SQE Practitioners Working Group to foster sharing of best practices, issues, templates, etc.
- Test Plans; continued development (and refinement) of the Code Test Plans as they relate to SQE issues.

## Conclusion

The assessment team feels that considerable useful information was gathered during the assessment, that there is a positive environment for SQE implementation, that code teams are currently implementing SQE “well” given resource constraints, and that there is a constructive path forward for the SNL ASCI Applications Program in the area of SQE. The assessment team feels that there is value in conducting future large-scale assessments (similar to the assessment documented here) as well as in conducting small-scale assessments. Large-scale assessments establish the overall project level of compliance with the SQE Practices. Large-scale assessments should be infrequent (approximately one per fiscal year). Two types of small-scale assessments should also be considered. First, a small-scale assessment involving a single code team can be performed as needed to assess that code teams progress on SQE. Second, a small-scale assessment can be performed to assess the implementation of a single process across multiple teams. Both of the small scale assessments can be valuable tools for helping code teams between large-scale assessments and can be conducted as frequently as needed.

## **Conduct and Scope of the Assessment**

The 2002 ASCI assessment was conducted following the process described in the *ASCI Apps Sandia Procedure (AASP) 13-1* [1]. A full program GAP assessment [2] was performed. The assessment started on 12/11/2001, and concluded 3/12/2002 at which time the assessment team provided a sponsor out-briefing. The majority of the assessment activities were conducted in a 7-week period beginning the week of Jan 21<sup>st</sup>, 2002, and concluding the week of February 25<sup>th</sup>. During this 7-week period twenty three of the twenty four codes teams were trained, performed their self assessment, were assessed, and were assigned scores. The other code team was assessed in an initial “pilot” assessment the week of December 11<sup>th</sup>. The scope of the assessment was ambitious and comprehensive as it covered every code targeted for weapons design or qualification by the ASCI Advanced Application program.

The name of the codes and their scores will not be publicly released. The scores will be associated with a code number, not the code name. Management made this commitment to foster a more open interchange between the code teams and assessors, because this was the first time many of the code teams were assessed, and because there was no time for the code teams to make any changes in their practices after publishing the *ASCI Applications Software Quality Engineering Practices* [2].

A number of issues are important to note with regard to the conduct and scope of the assessment:

- Other activities were dependent on the assessment results, so schedule was tight.
- The ASCI SQE Practices document was newly released at the time of the assessment.
- The assessment process was new.
- Simultaneous to the conduct of the assessment, code teams were receiving training on the SQE Practices document.

While not always the primary drivers, these issues were considered when addressing the assessment schedule.

## **Schedule**

A main goal of the assessment was to provide the AQMC with accurate information on the current status of software engineering practices by the end of Q2. Given this goal, the assessment schedule was established so that an evaluation could be performed and the assessment sponsor provided with timely feedback. The sponsor will provide input to the ASCI Quality Management Council (AQMC), which will then determine action plans to address process improvement. A sponsor out-briefing was conducted 3/12/2002, meeting this target schedule.

To determine the feasibility of meeting the assessment schedule, a pilot assessment of a single code team was conducted in December. The following 1-week schedule was proposed and a single code team selected for this assessment:

	Monday	Tuesday	Wednesday	Thursday	Friday
7:30	Training Session for Code Team A				
8am				Completion of interviews Review of Objective Evidence	
8:30	7:30-12:00 Noon				
9am	Bldg 880/Rm A16A		In-Briefing For Code Team A		Out-Briefing
9:30	Note: At least one member of A-Team involved in training session	Start of Assessment	Interview Sessions Review of Objective Evidence	Assessment Team Scoring Code Team on SQE practices	Code Team A
10am					
10:30		Self Assessment Due		Involved review of objective evidence and interview notes	
11am					
11:30					
12noon	Self-Assessment for Code Team A				
12:30					
1pm					
1:30	Note: Self Assessment activity involved code team mapping objective evidence to SQE practices and scoring themselves on each practice				
2pm					
2:30					
3pm					
3:30					
4pm					
4:30					
5pm					

**Table 1 - Pilot Assessment Schedule**

The assessment team for the pilot assessment consisted of two people. After the pilot assessment was completed, schedule and resource concerns were identified and addressed as described in the following paragraphs.

First, the initial schedule allocated insufficient time for the code team to collect and organize the required objective evidence. The code team was required to compile this information between receiving training and the start of the assessment. That period of time (1/2 day) was clearly inadequate.

Second, the pilot assessment team consisted of only two individuals. With only two assessors, it was not possible to adequately examine and evaluate the objective evidence provided by the code team in the time allotted by the schedule.

Third, accurate scoring of the code team on the forty six practices was a concern as the schedule allocated less than 10 minutes to review objective evidence and to score each of the forty six practices.

Finally, conducting the out-briefing on a Friday presented two problems. It left little time for the assessment team to prepare the out-briefing, and it was a burden to many of the technical staff due to the SNL 9/80 work schedule.

During the pilot assessment these issues were addressed by the assessment team working overtime. However, the assessment team felt that this was not a reasonable solution given the need to assess a total of twenty four code teams by the end of Q2.

After the pilot assessment the following changes were implemented.

To address the first issue, code teams were still scheduled to receive training on a Monday, however, their assessment would not start until Tuesday of the following week. This gave the code team a full week to assemble and map their objective evidence to the forty six practices. The assessment team felt that giving the code teams more time to collect and map their evidence to the practices in the SQE Practices document [2] would reduce the amount of time required for the assessment team to evaluate and score each practice.

To address the second and third issues, three additional members were added to the assessment team. All of the additional members were part-time. At this point the assessment team consisted of five assessors, three observers, and the site coordinator (see Roles).

To address the final issue the code team out-briefing was moved from Friday to Monday afternoon. The revised assessment schedule is shown below.

	Monday	Tuesday	Wednesday	Thursday	Friday
8am	Training Session for Code C				
8:30					
9am	Note: At least one member from assessment team providing training	In-Briefing for Code Team B	Assessment Code Team B (cont.)	Assessment Code Team B (cont.)	Assessment Team
9:30					Wrap-up of Scoring & Preparation of Out-Briefing for Code Team B
10am					
10:30			Start of Assessment for Code Team B	Interviews and Objective Evidence Review	Interviews and Objective Evidence Review
11am					
11:30					(for following Monday)
12noon	Note: Start of Self Assessment for Code Team C	Self-Assessment and Objective Evidence from Code Team B provided to Assessment Team			
12:30					
1pm					
1:30					
2pm	Out-Briefing given to Code Team A	at this time			
2:30					

3pm	(Assessed Previous Week)				
3:30					
4pm					
4:30		Note: Code Team B			
5pm		Trained Previous Monday			

**Table 2 – Revised Assessment Schedule**

## ***Critical Success Factors***

### **Sponsor Support**

A popular saying among quality professionals is “name the top 3 reasons why initiatives fail: management, management, management.” This is especially true in an assessment and the assessment team was fortunate to have an assessment sponsor who truly championed the process and provided the resources needed to complete the assessment on time.

Examples of this support include the dedication of a large high-tech conference room for multiple weeks, enthusiastic personal involvement, providing additional resources when they were needed, and dedication of time for the sponsor out-briefing. Because of the sponsor’s enthusiastic support, the code teams all understood that the assessment was a high priority.

### **High-performance Teams**

Given the large number of code teams being assessed (twenty four), the number of practices being reviewed (forty six), the compressed schedule, and the limited preparation time it was essential that the assessment team “gel” quickly into a high performance team.

The five assessors had to be very flexible about schedule, individual assignments, and accepting changes in the assessment schedule without missing a step. For example, team members automatically assumed responsibilities for one another when:

- team members were absent due to illness and accidents.
- team members were absent due to schedule conflicts.
- interviews scheduled for a single code team needed to be split into multiple (simultaneous) interviews of 2 code teams.

The assessment team was well positioned for high performance in part due to the team being composed of individuals with overlapping skills including:

- SQE
- SQA
- Auditing
- Computer Science

- Modeling and Simulation
- Project Management

A specific factor that contributed to the efficiency of the assessment team is that most of the assessment team members had previously worked together on similar assignments.

### **Site Coordination**

In any large assessment, the function of the site coordinator is extremely vital. The site coordinator is responsible for the logistics of the assessment (AASP 13-1 [1]). The site coordinator was responsible for scheduling interviews for all twenty four code teams and code teams managers. The site coordinator was also responsible for conference room logistics, production of copies of objective evidence for the assessors, collection of code team self-assessments, and other support. The site coordinator for this assessment performed all of these responsibilities extremely well.

### **Development of a Scoring Process**

Another important aspect of the assessment was a consistent approach to determining scores for each of the forty six practices both by the assessment team and the code team. There are some important aspects to note in this regard.

All code teams completed a self-assessment after attending the assessment training and prior to their code team in-briefing (Table 2 – Revised Assessment Schedule). The code teams’ self-assessments were based on the instructions given to them during the training.

The assessment team was trained on the assessment procedure prior to the beginning of the assessment. This included training on the scoring approach. Once the assessment team started interviewing and scoring code teams, it was quickly decided that the scoring approach was not adequate and an enhancement was implemented. The initial scoring approach allowed for the following scores:

- 0; indicating that the practice had not yet been included in any plans.
- 1; indicating that the code team planned to implement the practice
- 2; indicating that the code team was implementing the practice and could provide some objective evidence to support that contention.
- 3; indicating that the code team had fully implemented the practice.

Working with code teams it became apparent that more granularity was needed to differentiate between the levels of implementation. Another issue was that code team interpretations of the meaning of 0 were not consistent with the intended meaning.

The assessment team decided that a modified scoring approach would be needed in order to derive maximum value from the assessment. The assessment teams’ scoring of practices became the following:

- 0; indicating that the code team had no intention of implementing the practice.

- 1; indicating that the code team had objective evidence that the practice was being planned for implementation.
- 2; indicating that the practice was being implemented and that the code team had at least some objective evidence of implementation.
- 3; indicating that the practice was completely implemented, in a reproducible manner, and the code teams' practice document was complete.
- +/-; indicating a relative level of implementation within the numeric scores. The complete set of available scores became {0, 1-, 1, 1+, 2-, 2, 2+, 3-, 3}. The assessment team recommends that the SQE Practices Document [2] and the assessment procedure [1] be modified as needed to codify this scale for future assessments.

The assessment team recommends that the SQE Practices Document [2] and the assessment procedure [1] be modified as needed to codify this scale for future assessments.

The assessment team also had to decide how to handle issues associated with some practices whose definitions were less than clear and with inconsistent practice targets. An example of an unclear practice is practice 1a, which referred to software requirements. Almost every code team interpreted this practice as relating to system requirements instead of software requirements. Practice 2a (Derive the Design) is an example of inconsistent target scores. This practice had a target of 1 and yet, practice 2c (Document the Design) had a target of 3. Many code teams wondered how they could do a "3-level" job of documenting the design when they were only responsible for a "2-level" job of deriving the design.

### ***Roles and Responsibilities***

The assessment team consisted of 5 assessors, a site coordinator, and 3 observers.

#### **Assessment Team:**

•Mike Williamson	Co-Lead & Tech Specialist	Org. 6536
•Alex Treadway	Assessor & Tech Specialist	Org. 9519
•Harvey Ogden	Assessor & Tech Specialist	Org. 6536
•Laura Lang	Assessor & Tech Specialist	Org. 9515
•Kathleen Byle	Co-Lead & Tech Specialist	Org. 9519

#### **Site Coordinator:**

•Lora Bonano	Org. 9516
--------------	-----------

#### **Observers/Consultants:**

•Donna Eaton	Org 9519
•Molly Ellis	Org 9519
•Dwayne Knirk	Org 12326

## **Assessment Process & Results**

Following the pilot assessment, the remainder of the code teams were assessed over a period of 7 weeks. Including the code team evaluated during the pilot week, a total of twenty four code teams were assessed.

### **Assessment Process**

After all interviews were concluded for a specific code, the assessment team met, reviewed the objective evidence, and scored each practice. The scoring of each code team required between 2 and 4 hours. Prior to each code team out-briefing the assessment team completed scoring and team specific themes and suggestions. By the end of the assessment period, results had been generated for all twenty four teams.

For the purposes of this assessment report, the term “code team” refers to one or more ASCI codes for which the assessment team was provided a self assessment form. There were instances when two “ASCI” code teams submitted one self assessment form and for the purposes of the assessment received one set of scores because they shared implementation of the practices.

The results for all twenty four teams were consolidated into a single initial draft data set using Microsoft Excel. Using this data set the assessment team conducted analysis on the assessment results for the sponsor out-briefing. The initial draft data set was later verified and updated results generated. (a single typographical error was found). Results of the analysis include team specific plots as well as project average results (across all twenty four teams).

Various plots were generated both at the individual code team and program levels. Plots were selected for inclusion in this document based on the assessment sponsor’s request and on the assessment team’s analysis. The following plots are provided in this section:

- A program level plot showing assessment team and code team (self assessment) total scores. This plot was specifically requested by the assessment sponsor.
- A program level plot showing the average assessment team scores for all code teams minus the AQMC target scores by practice.
- A program level scatter plot of sorted team total scores, with the AQMC goal.
- Individual charts for the three code team exit questions demonstrating the good, the bad, and the opportunity.

Additional team level plots are provided in the appendix, specifically, plots for each team showing the assessment team scores minus the AQMC targets, and the assessment team scores minus the code team scores.

### **Assessment Team Scores and Code Team Scores**

It is useful to determine how well the overall assessment team scores for each team compare to the code team self-assessment scores. This provides insight into the overall

code team understanding of the SQE Practices document [2], code team training, and the assessment requirements.

Figure 4 shows the total assessment team score for each code team plotted next to the code team self-assessed score. To create this plot all practice scores were combined into a single score for each team. Both assessment team and code team totals were created for each code team. In twenty three of the twenty four cases the assessment team scores were lower than the code team self assessment scores. In some cases the assessment teams scores were significantly lower (~20%). Recall that the total target score was eighty seven.

The assessment team identified several recurring themes that help to explain the majority of the differences between the assessment team and code team scores:

- Lack of objective evidence. Code teams consistently scored themselves higher than the assessment team felt could be supported by the objective evidence that they provided.
- Lack of code team familiarity with the SQE Practices document [2]. This was to be expected as this document had only been released only a short time prior to the assessment.
- Assessment preparation. Code teams that did a good job of preparing and organizing their objective evidence generally scored higher than code teams that did not.
- Access to SQE expertise. Code teams that had been working with an SQE consultant, or had internal SQE expertise, or contributed resources to writing the SQE Practices document [2] scored higher than teams with limited SQE expertise. (e.g. The teams with the top 4 scores all fall into this category.)

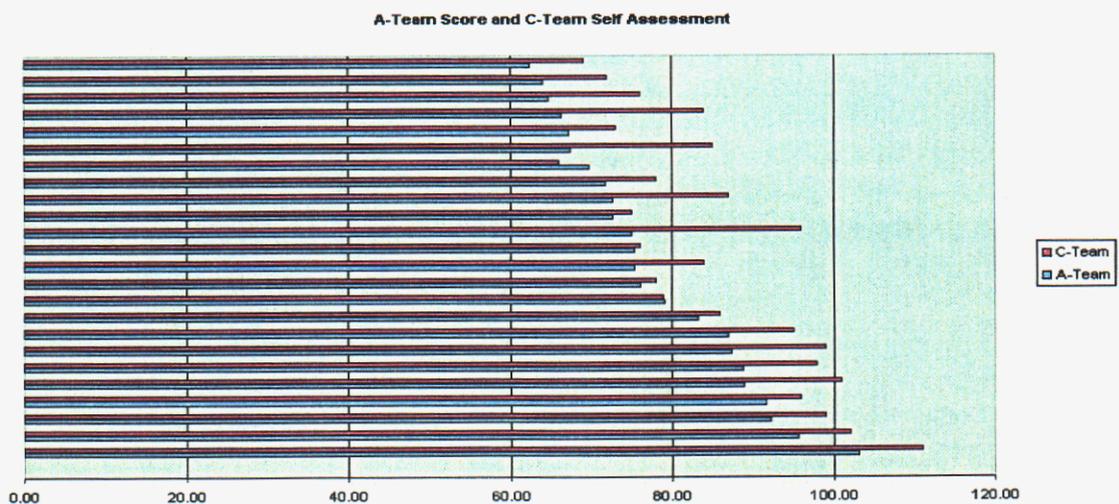


Figure 4: Code Team Self Assessment Score and Code Team Scores

Plots of the assessment score minus the team score for each practice for each team are shown in the Appendix.

Another useful program level measure is represented in the next plot. Figure 5 shows the average assessment team score minus the AQMC target score, by practice. Results **above the center line** indicate that, on average, teams are **exceeding the AQMC targets**. Results below the center line indicate that work will be needed to reach AQMC targets. Based on the AQMC targets for the forty six practices, twenty eight practice targets are, on average, being met and eighteen practice targets are not being met.

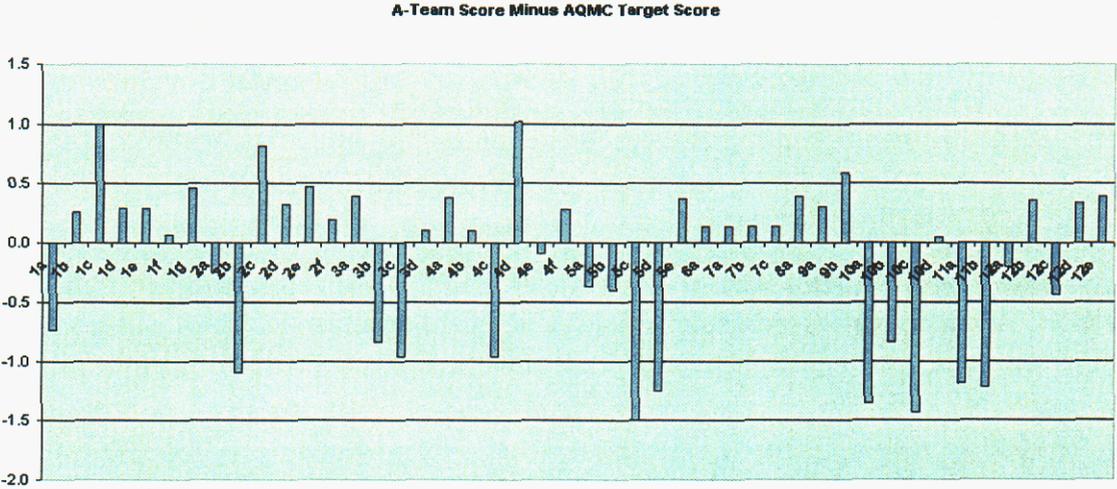


Figure 5 - Average Scores Minus Target Scores (Across all 46 Practices)

Another program level plot is provided in Figure 6. Total scores were calculated for all teams, the teams sorted by score, and the results graphed in a scatter plot. The AQMC total target score (eighty seven) is provided to show the AQMC total expected score. Sixteen of the teams fell below the AQMC target score and eight of the teams exceeded the target.

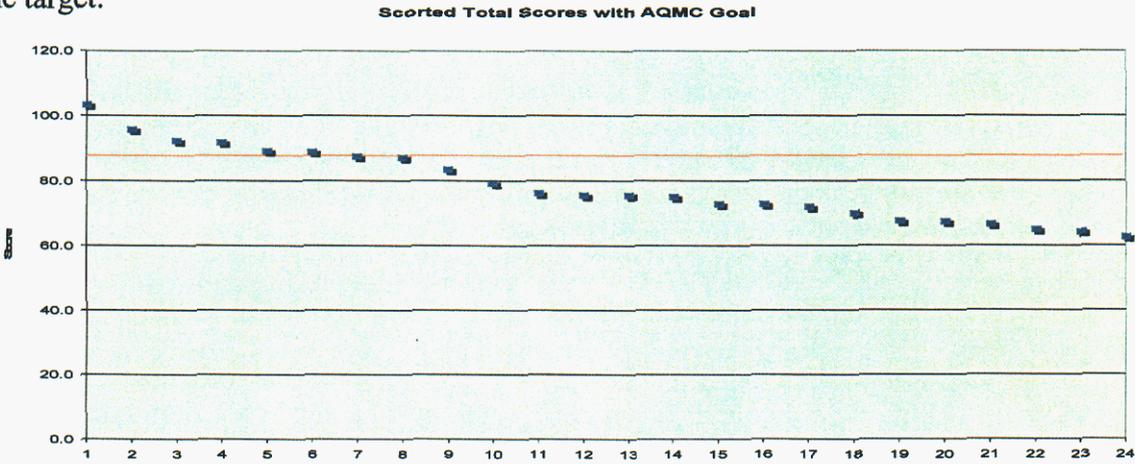


Figure 6 – Sorted Total Scores with AQMC Goals

Plots of the assessment score minus the target score for each practice for each team are shown in the Appendix.

## **Interview Exit Questions**

At the end of each assessment interview the interviewees were asked three questions. While the wording varied to some degree (due to different interviewers) the gist of the questions were:

- Other than “I work with good people” what works well in your working environment (i.e. what could you hold up as a good practice).
- What is not working well in your environment (i.e. what needs fixing).
- What would you like to see as an output from this assessment (i.e. you have invested time in this process, what outputs would help justify your time investment).

Each interviewee was allowed to provide as many comments to each question as they desired. All interviewees were told that the assessment team would accept additional comments later via email, phone, or personal contact. A small number of additional comments were received via those mechanisms.

The assessment team often referred to the three exit questions as “**the good**”, “**the barriers**”, and “**the opportunity**”. The following sections and charts summarize the responses to each of the three exit questions.

Examining the three most common responses for each question provides some interesting insights. First, note that the same two issues (SQE Tools and SQE/SQA Support) are in the top three responses for all three exit questions. In fact, many interviewees listed the same SQE Tools or SQE support issue as a positive and a negative example. Comments indicated a desire to stress that some support (tools and knowledge) is being provided but that what is being provided is not adequate.

### **The Good**

- Team Structure/Organization
- SQE Tools
- SQE/SQA Support

### **The Barriers**

- SQE Tools
- SQE/SQA Support
- Management

### **The Opportunity**

- SQE/SQA Support (5x more common than the next most common response)
- SQE Tools
- Management

Due to the assessment approach, the number of exit question responses recorded by each assessment team member varied. This was partially due to the lack of time for the assessment team to meet following every interview and agree on results of that interview, and partially due to the fact that not all assessment team members attended all interviews (due to schedule conflicts and team member illness).

The collection of exit question responses occurred as follows. During the interviews the assessment team members were asked to take notes and record comments. After all interviews were completed for a code team, each assessment team member was responsible for recording and then gathering all of their observations together and providing that information (in electronic format) to the assessment team sub-team responsible for response categorization. That sub-team met and reviewed all of the comments and observations submitted by each assessment team member. All comments and observations were treated equally.

Each of the three exit questions and the code team responses to them are discussed in more detail in the following sections.

## **The Good**

The first exit question asked for examples of what was working well. Code team members were cooperative and expressive and all interviews resulted in code teams providing several examples of what was working well in their environments.

Given adequate time and team resources it would have been preferable to have the assessment team meet immediately following each interview to consolidate their comments into a single set of data. This was not possible for this assessment due to the tight schedule. For this assessment, all comments and observations from each assessment team member were added to the exit question response list. No data reduction was preformed. The entire list was then reviewed and the data was grouped into the following six categories.

1. Team Structure
  - Small teams
  - Proximity (code team comments included co-located and remote physical location as examples of “good” team structure.)
  - Rapport
2. Communication
  - Email
  - Hallway discussions
  - Web (applications web sites, WFS, ...)
  - Regular team meetings
  - Bi-directional communication with customer
  - Communications with support teams (CSU)
3. Tools
  - SIERRA Framework

- SIERRA Tools
- ALEGRA/NEVADA
- Doxygen (a documentation tool)
- 4. Management
  - Direction
  - Leadership
  - Support
  - Proper tasking
  - Schedule management
- 5. SQE/SQA
  - Methodologies (code team comments included references to extreme Programming, Agile methodologies, pair programming, design via the Rational process, ...)
  - A defined process that the code teams can follow
- 6. SNL Organizational
  - Quality of people/resources
  - A pervasive belief in doing the right thing
  - A matrix structure that provides access to different resources

Figure 7 shows the total number of responses in each category.

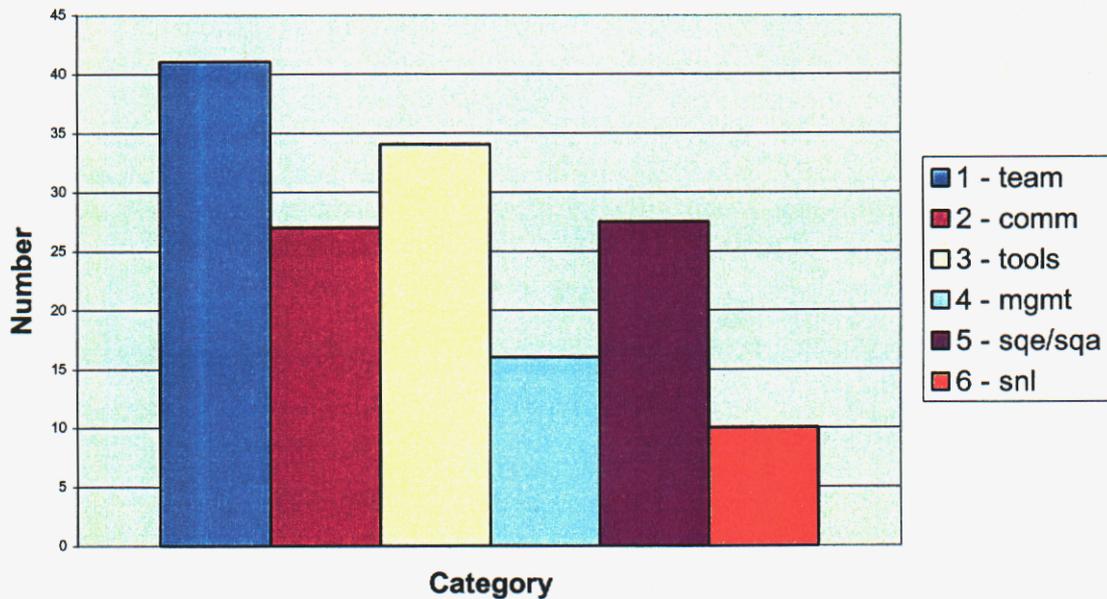


Figure 7 – Bar Chart of Code Team Responses to “The Good”.

The top three “good” areas of comment were teams, tools, and SQA/SQE.

## The Barriers

Teams were asked to identify what was not working in their environments and were able to provide a number of responses. Interestingly some of the items that showed up as “good” examples also showed up as “barrier” examples. For example teams referenced the framework tool group as providing valuable tools that are difficult to use, too limited, and not adequately documented.

As before, all responses were categorized. The six previous categories were again utilized and a seventh category was added.

1. Team
  - Too close together – hard to get work done (interruptions)
  - Too distant/remote – hard to get together
2. Communications
  - Hallway conversations not documented
  - Meetings not documented
  - Lack of access to customers
  - Customers not used to providing acceptance criteria
  - Lack of knowledge about sources of information
3. Tools
  - SQE
    - i. Lack of funding for SQE tools
    - ii. Need someone to evaluate and recommend SQE tools
    - iii. Some of the current SQE tools do not work well
  - Framework Tools
    - i. Slow
    - ii. Difficult to use
    - iii. Documentation is out of date
4. Management
  - Lack of consistent direction
  - Changing priorities
5. SQE/SQA
  - No time for SQE/SQA
  - Lack of trained SQE/SQA resources
  - SQE takes time away from the “real” work
6. SNL Organizations
  - Research more valued than S/W development (or QA).
  - “Trust me” mentality.
  - Politics
  - Matrix structure – people v. skills v. ramp time
7. Resources Issues
  - Inadequate funding
  - Access to people
  - Access to skill sets
  - Ability to get resources on schedule

- ASCII (big machine) CPU access

Figure 8 shows the number of responses in each category.

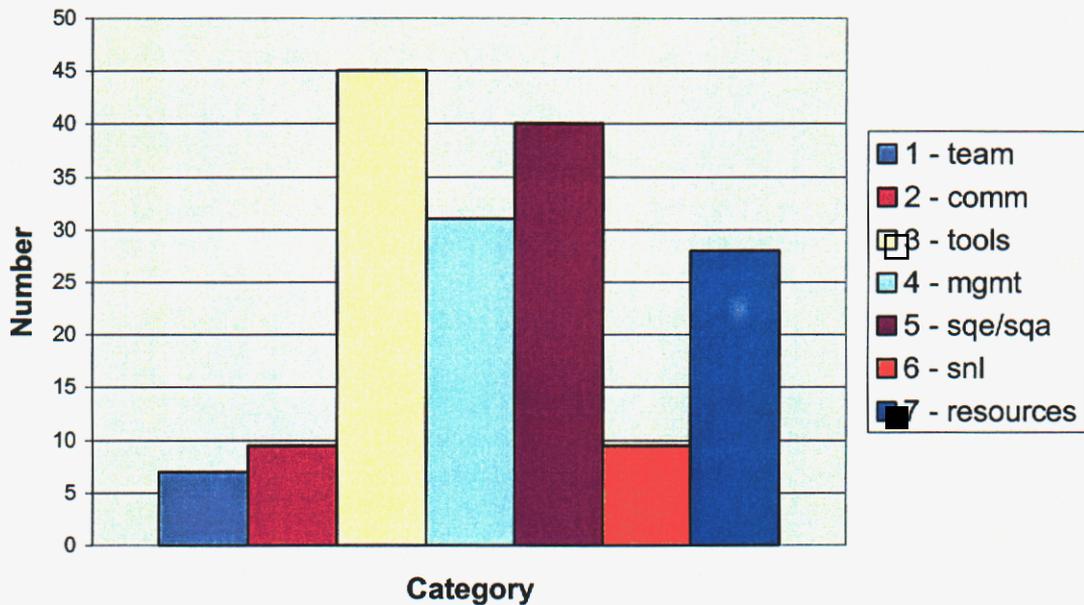


Figure 8 – Bar Chart of Code Team Responses to “Barriers”

The top 3 areas of comment for barriers were tools, SQE/SQA, and management.

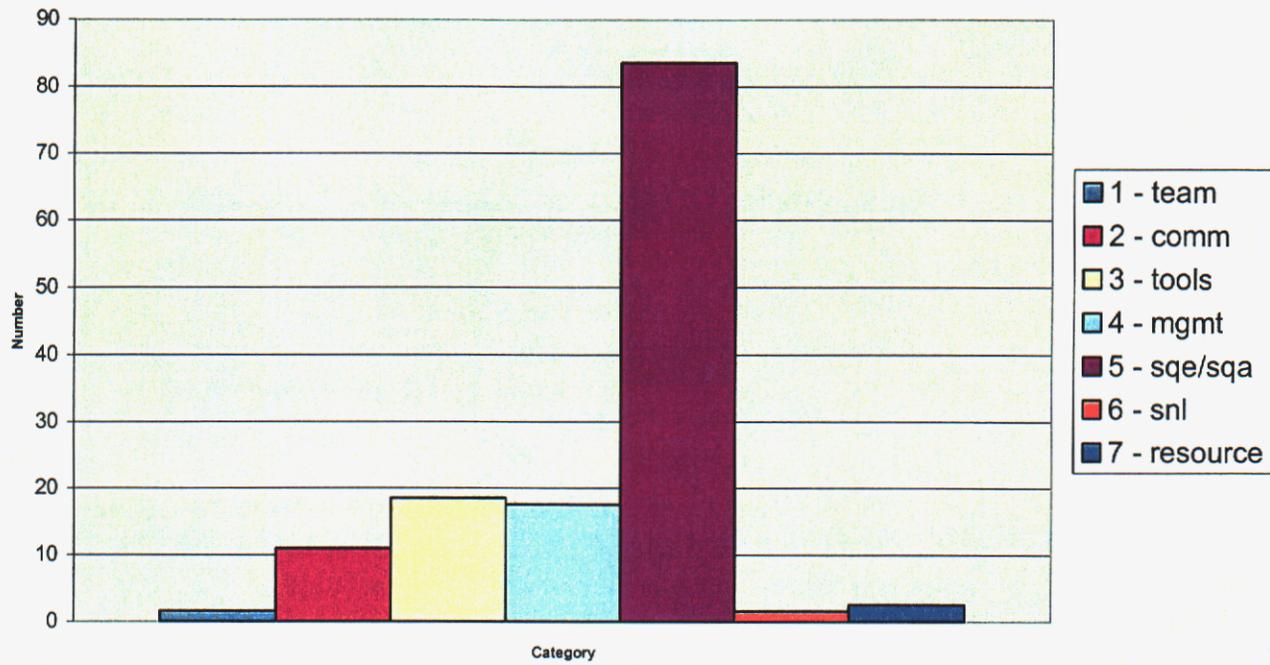
## The Opportunity

For the final exit question, interviewees were asked to comment on what they hope to come out of this assessment. Responses fell into the previously defined seven categories, with details as follows:

1. Team
  - Provide proof and examples on how SQE will help me do my job better
2. Communication
  - Why SQE is important to outsiders
  - Improve the Practices document
  - Team report on assessment results (this document)
  - Mentoring for best practices
3. Tools
  - Infrastructure for SQE
  - List of suggested tools

- Better process for releases
  - Automated tools
  - Supporting documentation in one place
  - Why some of the libraries and tools (SAF, ExodusII, SEACAS tools) were not included in the review?
4. Management
- Where should we be one year from now
  - Improve scheduling and dependencies between code teams
  - Educate customer and staff on their relative responsibilities regarding SQA (acceptance criteria)
  - Negotiate reasonable targets
  - AQMC needs to update practices and update target scores (as per their job description)
  - SQE practices need to be included as milestones
5. SQE/SQA
- Easily implemented improvements – identify areas where improvement can be easily achieved
  - Identify best practices
  - Set SQE/SQA priorities and direction
  - Provide examples
  - Train those selected for the external assessment
  - Provide results of the assessment – (report to the teams)
  - Resource allocation suggestions regarding SQE v. SQA
  - Explain benefits of SQE in this environment
  - Provide training on SQE practices
  - Provide training on “objective evidence”
  - Provide a baseline against which future improvement can be measured
6. SNL Org
- Too many requirements (BCP, ASCI, SNL Corp, ...), explain which one controls (what are they responsible for, what controls, ...)
  - ASCI records management
7. Resources
- SQE funding
  - SQE skilled personnel needed

Figure 9 contains the plot of responses to this question.



**Figure 9 – Bar Chart of Code Team Responses to “The Opportunity”**

The fact that the most common response to the “opportunity” exit question related to SQE/SQA issues should not be a surprise in an SQE/SQA assessment. What is interesting is that the next most common response, “SQE Tools”, followed closely by “management” were referred to by codes teams only 1/5 as often as SQE/SQA issues.

## **Better Practices**

During the assessment period the assessment team was on the lookout for best practices. In the context of this assessment, a best practice is one that represents an implementation of related practices (phases), or implementation of what the assessment team considered to be a key practice, in such a way that the implementation is effective, efficient, and reproducible. Both types of best practices are contextual to a specific code or environment.

The assessment team did not find any best practices, however the team did identify several “better practices”. A better practice is one that is not quite up to the level of a best practice, but can easily be elevated to best practice level with additional work.

In the following sections frequent reference is made to creating, enhancing, and documenting processes. It is important to know that the assessment team feels that a process must always add value. In the context of this assessment, a process that enables teams to achieve a better practice has the following attributes:

- effective process,
- efficient process,
- a process tailored to meet the needs of the code team,
- a process which provides value added to the final software product, and
- a process which provides for continuous process improvement.

Examples of code teams implementing the various practices close to or at the best practices level are provided for purposes of sharing of quality improving information among the code teams. These examples are not exhaustive; they do not reference every code team implementing a practice at the “better practice level”.

### **Requirements Phase (Practices 1a – 1g)**

**Requirements Management:** “the purpose of this subphase is to develop, capture, baseline, and communicate product requirements that are to be implemented as software.” [2] Although the assessment found that a number of code teams were not clearly distinguishing between system requirements and software requirements, there were examples of teams very close to implementation of the requirements phase at a better practice level.

Among the top scoring code teams, the strongest practices in the requirements phase were 1a, 1b, and 1c. These practices cover deriving software requirements, documenting software requirements, and assessing the feasibility of software requirements including projecting necessary resources to implement the requirements.

No code team was consistently strong in the area of determining links to other layers (1d) and/or ensuring requirements traceability throughout the subsequent software phases (1e).

Teams scoring high on practices 1a, 1b, and 1c had processes defined for the derivation and management of requirements at the team level. Requirements were tracked using a

tool such as Microsoft Project™, and were further broken down into an appropriate level of granularity.

Examples of requirements “better practices” observed during the assessment include:

- Several code team managers held regularly scheduled meetings (often offsite) with customers to discuss requirements. The meetings included a presentation of the current requirements set and included elicitation requests for new requirements. Meetings could include discussion of appropriate requirements based acceptance criteria. This activity supports practices 1a, 1b, and 1d. This activity could be moved from a better practice rating to a best practice rating by:
  - Collecting (and version controlling) requirements discussions, decisions, and concurrence (customer, technical, quality, management) made as a result of the meetings.
  - Establishing, or initiating the establishment of acceptance criteria that can be used to demonstrate the successful implementation of the collected requirements. Specifically in order to achieve a best practice, a requirement should not be fully accepted until the acceptance criteria for the requirement has been defined.
  - Customer concurrence of the acceptance criteria would also be extremely valuable.
  
- Code Team 2, 5, 6, and 8 have implemented practices 1a, 1b, and 1c in a “better practices” manner. The teams have a requirements process that is well defined and reasonably complete. This process could become a best practice once acceptance criteria (1d), link definition (1e), and requirements tracability (1f) are further addressed.

### **Development: Design Subphase (Practices 2a – 2f)**

**Design:** “the purpose of the Design Subphase is to describe components in a manner that can be implemented in software .” [2]

Examples of design related “better practices” observed during the assessment include:

- Code Team 1 implemented several design practices in a “better practice” manner.
  - Communicating the design (2b) using group presentations, specific design and refactoring meetings, bi-weekly team meetings, and web based team areas. To become a best practice the critical design decisions that are made as a result of presentations and meetings should be documented and traceable.
  - Documenting the design (2c) using the tool TogetherSoft Control Center. Note: other tools could also be used in a best practice manner for design documentation.

- Code Team 12 used meetings, email and phone conversations to derive and communicate the design to the team. The team maintains design notes that document the decision points.

### **Development: Implementation Subphase (Practices 3a – 3d)**

**Implementation:** “the purpose of the Implementation subphase is to transform the software design into code.” [2]

The activities that occur during implementation involve: evaluating the impact of implementation to design and requirements (3a), translating the design into code and other artifacts (3b), communicating issues with the team (3c), and reviewing and approving implementation artifacts (3d) [2]. It should be noted that an assumption is that code teams are producing code; that alone does not constitute a better practice. Maintaining the needed links (back to requirements and design for example) is a key for the achievement of software quality. In general, the link from design to implementation was not demonstrated by the objective evidence. This lack of objective evidence made it difficult to determine if software requirements were being translated into code. It was obvious that a number of code teams were in an iterative design-code-test cycle; however this process was not formally defined and documented and reviews not specified. In discussions with the code teams it was apparent that many informal reviews were taking place. By describing the iterative design-code-test approach with appropriate links and reviews, many of the code teams could move very close to being at the better practice level for this subphase.

- Code Team 3 achieved a better practice’s level in the activity of communicating issues with the team. In this practice, it is important that issues relating to implementation are identified, assigned to individuals, and tracked to resolution. Team 3 held regular meetings, generated minutes of these meetings, used a web based tool (enotebook) and an issue tracking tool (Mantis). Note that other tools could be used to achieve a better practice in 3c. However, this process and tool set was efficient and effective for this particular team.

### **Development: Test Subphase (Practices 4a – 4f)**

**Test:** “the purpose of the Test Subphase is to identify defects in the software product and to demonstrate that the software product meets its software requirements.” [2]

Examples of “better practices” observed during the assessment include:

- Code Team 1 has implemented the test subphase in a better practices fashion. A version controlled test plan is maintained. In addition, tier 1 (tests with exact analytical solutions), tier 2 (tests with semi-analytic solutions), and tier 3 (idealized problems suitable for code comparison exercises) test cases [5] are identified, nightly builds and regression testing are conducted on all target platforms, nightly test results are emailed to team members, and a process exists for reviewing test results. To become a best practice, the tier 2 and tier 3 test description documents need to move from draft to published status and the scope

of testing needs to expand to include adequate tier 2, tier 3, and release acceptance tests.

- Code Team 2 has implemented a nightly test script that exercises a group of test cases and documents the results. Fast and slow test suites are available and are run on a scheduled basis. All code updates are required (by procedure) to include unit tests for new features and functionality. To become a best practice the documentation of the test system needs to be brought up to date and the ease of use of the test tool set must be improved.

### **Release Phase (Practices 5a – 5e)**

**Release:** “the purpose of the Release Phase is to manage a production version of the software product that is distributed to customers.” [2]

Examples of release phase “better practices” observed during the assessment include:

- Code Team 1 and Code Team 2 have a similar release process. The process describes how releases are scheduled, planned, and managed. The process specifies roles and responsibilities, mechanisms for requesting releases, types of releases, the promotion model, reviews and approvals, and release tracking. To become a best practice both processes need be streamlined, finalized, and released.

### **Project Management (Practice 6a)**

**Project Management:** “the purpose of Project Management is to ensure that adequate funding and resources are available to allow successful completion of deliverables and required software practices .” [2]

Practice 6a relates to the program level more than to the code team level. While this practice has been implemented by team management the evaluation of the practice was rather binary (implemented or not implemented) and did not lend itself to identification of better practices.

### **Tracking and Oversight (Practices 7a – 7c)**

**Tracking and Oversight:** “involves the tracking and reviewing of projected accomplishments and results with respect to how they are described in the project plan.” [2]

Practices 7a – 7c relate to the program level more than to the code team level. While these practices have been implemented by team management the evaluation of these practices was rather binary (implemented or not implemented) and did not lend itself to identification of better practices.

## **Risk Management (Practice 8a)**

**Risk Management:** “involves identifying, addressing, and mitigating sources of risk before they become threats to the successful completion of a project.” [2]

Practice 8a relates to the program level more than to the code team level. While this practice has been implemented by team management the evaluation of the practice was rather binary (implemented or not implemented) and did not lend itself to identification of better practices.

## **Support Elements (Practices 9a - Practice 12e)**

**Support Elements:** “the purpose of Support Elements is to help monitor and correct project plans against performance, conduct reviews of artifact content, train software developers, and document and preserve the results of the project” [2]

While some of the support element practices have been implemented by most of the code teams (esp. configuration management) no better practices were observed during the assessment.

## **Assessment Recommendations**

The assessment team has provided two types of recommendations relating to SQE (and other) improvements. The first type is “easily implemented improvements” meaning those activities/tasks/objectives that can be achieved with little or no cost and can be implemented in a short time frame. The second type is “other opportunities” meaning those activities/tasks/objectives that will require budget and/or more time to achieve. The assessment team has identified items in both categories.

## ***Easily Implemented Improvements***

For a recommendation to be an easily implemented improvement it must not require any significant budget or schedule modifications. Implementation must be quick and relatively easy to achieve. Under this category the assessment team identified several opportunities.

## **Documenting Quality Improving Work**

The assessment team was often provided with little objective evidence for those routine activities that result in important decisions. Such activities usually take place on a day-to-day basis, for example in team meetings, hallway discussions, emails, and phone calls. The shortage of this kind of documentation was particularly evident for practices in the requirements, design, and implementation phases.

The failure to collect this information is more than a documentation issue; it can be **quality improving**. If this information is not documented, it is difficult to track key decisions to completion and it is likely that some issues will be “lost”. Without documentation, it will be difficult to recall the reasoning behind certain design or implementation decisions. This is especially true if there is turnover of key personnel on the team or if the technical environment is very complex (e.g. coupled codes).

A simple and low cost solution is easily implemented.

1. Meeting/discussion notes should be generated for all project related meetings. The contents of these notes should include (at a minimum):
  - Date and time and location of the meeting
  - Attendees
  - Important decisions and concurrence on these decisions
  - Important new assignments
2. Important emails should be saved in a retrievable manner.
3. If a phone call or hallway discussion results in an important decision, it should be documented (e.g. by email) and distributed to the entire team.
4. All important artifacts should be under configuration management and version control. This can be as simple as a project notebook containing a printed copy, or it could be Web FileShare, or some other tool with configuration management capability.
5. This information should be readily assessable to the entire team. Many teams maintain a web site that include electronic versions of meeting notes, design

notes, and other objective evidence. The method each team uses for sharing this information should be clearly communicated, understood, and used by all team members.

## **SQE Resources**

Teams with SQE knowledge (or access to SQE consulting) implemented the SQE practices [2] more consistently and thus scored higher in the assessment. With access to SQE resources there should be fewer negative SQE issues.

While some teams have one or more team members with significant SQE experience or knowledge, some teams have very little. A solution to this issue is already being implemented as some teams have utilized various pockets of SQE knowledge at SNL.

It is recommended that the pools of SNL SQE consulting resources be identified so that they can be made available to all code teams. This recommendation only addresses the identification of existing resources, it does not address budget issues associated with the use of SQE consultation as the assessment team was told by several teams that they (the code teams) had budget for SQE resources, but lacked resources.

## **Issue Tracking**

All ASCI Application code teams should have access to an Issue Tracking tool for use in documenting and tracking bugs, issues, and enhancements. A web based issue tracking tool should be considered. A few code teams are using a “no-cost” web based tool called Bugzilla. Code Team 4 has been very proactive in evaluating issue tracking tools. Code Team 4 is currently using PVCS Dimensions and is assessing another issue tracking tool that is part of a collaboration toolset (SourceForge™). Other “tools” being used today include spreadsheets and Mantis.

## **Metrics**

Metrics (beyond what was generated during this assessment) are a necessary part of process improvement. Metrics can be used to provide insight into the “goodness” of software products and of the SQE practices used to develop the software products. Metrics provide insight into what is needed and when it is needed.

A prerequisite to generation of metrics is the collection of metric data. Collection requires that you have the ability and the need to collect requisite data. There are numerous potential metrics that might be of value to the code teams and to the ASCI Applications manager. Examples include estimated vs. actual effort for development, projected vs. actual budget, programming metrics, etc. However, prior to collecting data for metrics, it is essential that a decision be made as to what metrics are valuable to the project and how those metrics will be used.

The easily implemented improvement here is to identify the types of metrics that are of value (e.g. assessment results), to determine how these metrics will be used, and to decide how to collect data.

## ***Other Opportunities for Improvement***

### **Program Issues**

This assessment uncovered several opportunities for improvement in the quality program that could be addressed at the AQMC level. These issues and recommendations are described below.

### **Issue Tracking**

It is recommended that issue tracking at the program level be implemented. This will allow the AQMC's issues to be identified, tracked, and resolved. This is a necessary component of oversight of a quality program. For example, opportunities for improvement identified in this report could be tracked.

### **SQE Training**

Code teams that have some exposure to SQE knowledge, such as those working with someone involved with the SQE Practices document (SIERRA, SIERRA Tools, Xyce), did much better than teams without such exposure. That exposure established a foundation for the code teams to better benefit from the training that was provided.

It is therefore recommended that SQE training be conducted for all code developers on at least a yearly basis. Training needs to target increasing the overall awareness and knowledge of SQE practices and requirements. The SQE training that was offered this year was a good start [4], but it should be enhanced to make it even more useful to the code teams. The training should not focus merely on describing the requirements, but also on how to meet them, what resources/tools are available, templates and examples, and other practical suggestions.

### **SQE Resources**

SQE resources are in fairly high demand. The identification of available SQE resources (esp. pools of resources) would help the ASCI applications code teams find the help that they need. However, some teams do not currently have available funding to take advantage of these resources. It is recommended that providing this funding at the code team level become a project priority.

### **SQE Tools**

It is desirable that code teams to have access to a set of SQE tools and to an infrastructure to will help enable the teams to achieve the targeted SQE results. The AQMC needs to determine how this need will be addressed at the program level.

### **Records Management**

Program wide records management is needed. There needs to be a way of storing and retrieving important project records that is independent of the code project. This needs to

be easily assessable by appropriate personnel. It is a significant risk to depend solely on individual codes teams for the retention of vital project records.

## **AQMC**

**Deployment Document [2]:** Practice 1a (requirements) needs to be expanded to explain that system requirements and software requirements are not the same thing. Both system and software requirements must be gathered and tracked. This could also be accomplished by adding a practice before the current practice 1a, addressing the activities associated with capturing system requirements, followed by the derivation of those system requirements that will be implemented as software requirements.

For assistance in the overall assessment process, it is recommended that consistent numbering of the practices be followed through the deployment document. For example, in the text of the deployment document, the practices are not numbered. When going from the checklist to the deployment document, finding the description for a practice of interest was awkward for both the code and assessment teams.

Practices 6a and 8a do not cover all of the necessary aspects of project management, and risk management. Practice 6a currently addresses only project management at the program level. Project management at the code team level will become a quality improving activity as the ASCI applications move towards an increasingly coupled environment. Similarly, practice 8a currently addresses risk management at the program level. Risk management at the team level has the potential to have significant quality and schedule impact.

The assessment team expanded the score values to provide greater granularity. The AQMC should consider adopting the expanded scale.

**Program Infrastructure:** The AQMC needs to consider the development and distribution of additional infrastructure that may be needed to support the SQE effort. One example of an infrastructure process/procedure (that is currently being issued) is the assessment procedure AASP 13-1 [1].

**Distribution of Quality Documents:** Issuing a quality document as a SAND report is important as the SAND report establishes the credibility of the procedure, and management commitment to its implementation. Changes to a quality procedure are an inevitable result of process improvement, as the procedure is implemented, as assessment are conducted, etc. However, SAND reports are not easily modifiable. Changes to the procedure should be timely so that process improvement occurs as soon as possible.

It is recommended that, for those quality procedures which fall under the responsibility of the AQMC, such as the SQE practice's document, that the AQMC consider an additional

distribution method (i.e. web page) with a streamlined approval process for minor changes.

**Assessment Process:** It is recommended that the AQMC revise and issue AASP 13-1 based on lessons learned from this assessment. For example, provide guidance on detailed assessment scheduling, provide templates for use by assessors, additional guidance on conducting interviews, and appropriate use of expert judgment in scoring. In addition the SQE Practices Document [2] should be modified to specify that a process such as AASP 13-1 is needed to conduct an assessment.

The AQMC should re-evaluate and revise the target scores assigned to the forty-six practices for both consistency and attainability. For example, practice 4a “finalize test plan”, has a target score of 1. However, practice 4c “review test case outputs using acceptance criteria defined in test plan” has a target score of 3. It should be noted that some code teams indicated that they lack the resources (time, skills, etc.) to achieve the current AQMC target scores.

Each artifact produced as part of compliance with the practice document is required to be subject to three types of review’s; technical, QA, management [2, page 18, next to last sentence]. The SQE Document [2] specifically lists a review practice for phases 1, 2, 3, and 4 but not the other phases. There is no discussion of the relative value of each type of review. Scoring the review practices as currently written is awkward. For example, in the testing phase, the test plan may have all three required reviews, yet the test output may have only received a technical review. How then would the assessment team score the practice 4f, Review and approve Test Sub phase Outputs. It seems as if a separate review practice for each phase is redundant, as the three reviews should be clearly required for every artifact produced by following the practices in order to achieve a 3.

**Assessment Frequency:** The assessment team feels that there is value in conducting future large-scale assessments (similar to the assessment documented here) as well as in conducting small-scale assessments. Large-scale assessments establish the overall project level of compliance with the SQE Practices. Large-scale assessments should be infrequent (approximately one per fiscal year). Two types of small-scale assessments should also be considered. First, a small-scale assessment involving a single code team can be performed as needed to assess that code teams progress on SQE. Second, a small-scale assessment can be performed to assess the implementation of a single process across multiple teams. Both of the small scale assessments can be valuable tools for helping code teams between large-scale assessments and can be conducted as frequently as needed.

## **Code Teams**

**SQE:** The Formation of an informal SQE Practitioners Working Group is recommended. This would be a group comprised of the SQE owners, and/or practitioners from each code team, and SQE resources. This group would meet on a regular basis to discuss ASCI

Applications related SQE issues, best practices, tools, templates, etc. This would provide a forum for the sharing of SQE practices amongst the code teams and result in greater uniformity of SQE quality. Many “Better Practices” are already in use by various ASCI code teams. Leveraging these practices by sharing them with the other code teams would be a cost effective way of increasing quality. This working group would be one possible way of accomplishing this. This working group should have appropriate support from the AQMC.

**Test Plans:** It is strongly recommended that teams continue the development of their test plans. Test plans are vital in describing the overall verification strategy, what types of text case are executed, what events trigger the execution of test cases, what defines the success of a test (acceptance criteria), etc.

## References

1. *ASCI Applications Sandia Procedure (AASP) 13-1. Draft*, K Byle, M. Williamson.
2. *ASCI Applications Software Quality Engineering Practices*, SAND2002-0121, Jan 2002.
3. *ASCI Software Quality Engineering: Goals, Principles, and Guidelines*, SAN2001-0253P, Jan 2001.
4. *ASCI SQE Assessment Training Report. Draft*, Molly Ellis and Donna Eaton Org. 9519, March 2002.
5. *Guidelines for Sandia ASCI verification and validation plans*, SAND2000-3101, Pilch, Martin M.; et. al.

## Appendices

## **Summary of Suggested Changes for the SQE Practices Document [2]**

<b>Description</b>	<b>Reference in this Report</b>	<b>SQE Document Impact</b>
Provide a definition for “Best Practice”. Please consider our definition (page 25).	Page 25	Page 13, 41
The assessment team expanded the score values to provide greater granularity. The AQMC should consider adopting the expanded scale.	Pages 13-14, page 33	Section 3.3, page 19
Consider moving the target scores out of the SQE Practices document as the target scores will need to be changes more often than the document.	Page 33 (discussion on document types)	Page 44-49
Practice 1a (requirements) needs to be expanded to explain that system requirements and software Requirements are not the same thing.	Page 14, 25, 33	Section 3.3.1, page 20-21
Practice 2a (Derive the Design) is an example of inconsistent target scores. This practice had a target of 1 and yet, practice 2c (Document the Design) had a target of 3. Many code teams wondered how they could do a “3-level” job of documenting the design when they were only responsible for a “2-level” job of deriving the design.	Page 14	Page 45
For assistance in the overall assessment process, it is recommended that consistent numbering of the practices be followed through the deployment document. For example, in the text of the deployment document, the practices are not numbered. When going from the checklist to the deployment document, finding the description for a practice of interest was awkward for both the code and assessment teams.	Page 33	Multiple locations, for example page 20 and page 44.
Practices 6a and 8a do not cover all of the necessary aspects of project management, and risk management. Practice 6a currently addresses only project management at the program level. Project management at the code team level will become a quality improving activity as the ASCI applications move towards an increasingly coupled	Page 33	Section 3.4.1 and section 3.4.3

Description	Reference in this Report	SQE Document Impact
environment. Similarly, practice 8a currently addresses risk management at the program level. Risk management at the team level has the potential to have significant quality and schedule impact.		
The need for artifact review is not consistently represented. Page 18 specifies three types of artifact reviews that are required for all artifacts. Reviews are included as practices in some phases but not all phases.	Page 34	Page 18, 21, 24, 26, 30, 32, 36, 40
The checklist is not sufficient in itself to perform an independent assessment. It is an excellent tool for teams to perform self-assessments to gauge their level of software quality engineering. A paragraph should be added to the checklist section, recommending that an approved assessment process (such as AASP 13-1 or QC-1) be followed when conducting and independent assessment.	Page 34	Page 41

## ***First Time Assessment Lessons Learned***

The assessment team would like to document (and reinforce) some of the lessons learned (and lessons verified) during this assessment.

1. Management support is critical. Without vocal and visible management support the assessment will most likely fail.
2. Assessors must be independent
3. An assessment plan and assessment schedule must be developed.
4. Selection of the assessment team is critical. As a group the assessors must provide complete coverage of the subject area for the assessment. At least some of the team must have assessment experience. At least some of the team must have hands-on experience in the subject area.
5. Assessors must be given time immediately after interviews so that they may come to consensus on interview content and artifact quality while issues are still fresh in their minds.
6. Assessors must be given time to review evidence prior to conducting interviews. This also implies that evidence must be provided to the assessment team well in advance of the interview.
7. If conducting a large-scale assessment (many teams or many practices) consider a prototype approach where a single team is assessed and the assessment procedure is evaluated (and enhanced if needed) prior to conducting the large-scale assessment.
8. The assessment coordinator role is vital. This role coordinates schedules and facilitates issues for the assessment team. The assessment coordinator is responsible for getting the right people to the proper place, at the correct time, with all required documentation. All of this is needed if the assessment team is to do a proper evaluation.
9. When possible, members of the assessment team should be included in the team training sessions. This helps to ensure consistent representation of issues.
10. A checklist is not sufficient in itself to perform an independent assessment. It is an excellent tool for teams to perform self-assessments to gauge their level of software quality engineering. A paragraph should be added to the checklist section, recommending that an approved assessment process (such as 13-1, QC-1) be followed when conducting an independent assessment.

# Assessment Team Scores

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	
	1a	1b	1c	1d	1e	1f	1g	2a	2b	2c	2d	2e	2f	3a	3b	3c	3d	4a	4b	4c	4d	4e	4f	5a	5b	5c	5d	5e	6a	7a	7b	7c	
Code 1	2.3	2.7	2.3	2.0	2.0	1.7	1.7	2.3	2.7	2.7	1.0	3.0	2.0	1.7	2.3	2.0	1.3	2.7	3.0	3.0	2.0	3.0	2.0	2.7	2.3	2.3	2.0	1.7	3.0	3.0	3.0	3.0	
Code 2	3.0	3.0	3.0	1.0	2.0	1.3	2.0	2.0	2.0	2.0	2.0	2.0	1.3	2.0	2.3	3.0	1.0	1.3	2.7	2.3	2.7	2.3	1.0	2.3	2.3	2.3	2.3	2.0	3.0	3.0	3.0	3.0	
Code 3	2.3	2.3	2.0	3.0	2.0	1.0	2.0	1.0	2.0	1.0	2.0	1.3	1.0	2.0	2.0	3.0	1.0	2.0	2.0	2.0	2.0	2.0	1.7	3.0	2.0	2.0	2.0	2.0	3.0	3.0	3.0	3.0	
Code 4	2.3	3.0	2.0	1.0	1.0	1.0	2.0	1.7	2.0	2.0	1.7	2.0	1.0	1.7	2.0	2.7	2.0	1.0	2.0	2.0	2.3	2.3	2.0	2.0	2.0	2.0	2.0	2.0	3.0	3.0	3.0	3.0	
Code 5	3.0	3.0	3.0	1.0	1.3	1.3	1.7	2.0	2.0	2.0	2.0	1.0	1.0	2.0	2.3	2.0	1.0	1.3	2.7	2.3	2.7	2.3	1.0	2.0	2.0	1.0	2.0	1.0	3.0	3.0	3.0	3.0	
Code 6	3.0	3.0	3.0	1.0	1.3	1.3	1.7	2.0	2.0	2.0	2.0	1.0	1.0	2.0	2.3	2.0	1.0	2.0	2.7	2.3	2.7	2.3	1.0	2.0	1.0	1.0	2.0	1.0	3.0	3.0	3.0	3.0	
Code 7	2.0	2.0	2.0	1.7	1.3	1.0	1.3	2.0	2.0	1.7	2.0	1.7	1.7	2.0	2.0	2.0	1.0	1.7	2.0	2.0	2.0	2.0	1.0	2.0	2.0	2.0	2.0	2.0	3.0	3.0	3.0	3.0	
Code 8	3.0	3.0	3.0	1.0	1.3	1.3	2.0	2.0	2.0	2.0	2.0	1.3	1.0	1.3	2.3	1.7	1.0	1.0	2.3	2.0	2.0	1.7	1.0	2.0	2.3	2.0	2.0	1.0	3.0	3.0	3.0	3.0	
Code 9	2.0	2.3	2.3	1.0	1.3	1.0	1.0	2.0	2.0	2.3	1.7	2.3	2.0	2.0	2.3	3.0	1.0	1.0	2.3	2.3	2.0	2.0	1.7	1.0	2.0	2.3	2.0	2.0	1.0	3.0	3.0	3.0	3.0
Code 10	2.0	2.0	2.0	1.0	1.7	1.0	1.7	2.0	2.0	1.7	1.0	1.0	1.0	1.0	2.0	2.0	1.0	1.0	2.0	2.0	2.0	2.0	1.0	1.0	1.0	1.0	1.0	0.7	3.0	3.0	3.0	3.0	
Code 11	2.3	2.0	2.0	1.3	1.3	1.0	1.7	1.0	2.0	2.7	1.0	1.3	1.0	1.0	2.3	1.7	1.0	2.0	2.0	2.0	2.0	2.0	1.0	1.0	2.0	2.0	2.0	3.0	2.0	3.0	3.0	3.0	
Code 12	2.0	2.0	2.0	1.0	1.0	1.0	1.3	2.7	2.7	2.3	1.3	1.3	1.3	1.3	2.3	2.0	1.3	1.0	2.0	2.0	2.0	2.0	1.0	1.0	2.0	1.0	1.0	1.0	1.0	3.0	3.0	3.0	3.0
Code 13	2.0	2.0	2.0	1.0	0.7	0.7	1.0	2.0	2.0	2.0	1.0	1.0	1.0	1.0	2.0	2.0	1.0	1.0	2.0	2.0	2.0	2.0	1.0	0.7	1.0	1.0	1.0	0.7	3.0	3.0	3.0	3.0	
Code 14	1.7	1.0	2.0	1.0	1.0	1.0	1.0	1.0	1.3	1.0	1.0	1.0	1.0	1.0	1.7	1.3	1.0	1.7	2.0	2.0	2.0	2.0	1.7	1.7	2.0	2.0	1.0	2.0	3.0	3.0	3.0	3.0	
Code 15	2.3	2.0	2.0	2.0	1.0	1.0	1.7	2.0	1.7	1.3	1.0	1.3	1.0	1.0	2.0	1.7	1.3	1.0	2.0	2.0	2.0	1.0	1.3	0.7	1.0	1.0	1.0	1.0	3.0	3.0	3.0	3.0	
Code 16	2.3	2.0	1.0	1.0	1.0	1.0	1.0	2.0	2.7	2.0	1.0	1.0	1.0	1.0	2.3	2.0	1.0	1.0	2.0	2.0	2.0	2.0	1.0	2.0	1.0	1.0	1.0	1.0	3.0	3.0	3.0	3.0	
Code 17	2.0	2.7	2.0	2.0	1.3	1.0	1.0	2.0	2.0	2.0	1.0	1.7	1.7	1.0	2.0	2.0	1.0	1.0	1.7	1.0	1.3	1.3	1.0	0.7	1.0	1.0	1.0	0.7	3.0	3.0	3.0	3.0	
Code 18	2.0	2.0	1.7	1.0	1.7	1.0	1.0	1.7	1.7	1.0	1.3	1.3	1.0	1.0	1.7	1.7	1.0	1.0	1.7	1.7	1.7	1.7	1.0	0.7	1.0	1.0	1.0	0.7	3.0	3.0	3.0	3.0	
Code 19	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	2.0	1.0	1.0	1.0	1.0	2.0	2.0	1.0	2.0	2.0	2.0	2.0	2.0	1.7	1.0	1.0	1.0	2.0	1.0	3.0	3.0	3.0	3.0	
Code 20	1.0	1.0	1.0	0.7	0.7	0.7	1.0	0.7	1.0	1.0	0.7	0.7	0.7	1.0	2.0	2.0	1.0	1.0	1.0	2.0	1.0	2.0	1.0	2.0	2.0	2.0	3.0	2.0	3.0	3.0	3.0	3.0	
Code 21	2.3	2.0	1.7	1.0	1.0	1.0	1.0	1.7	1.0	1.0	1.0	1.3	1.0	1.0	2.0	1.0	1.0	1.0	2.0	2.0	2.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	3.0	3.0	3.0	3.0	
Code 22	2.0	2.0	1.0	1.0	1.0	0.7	1.0	1.0	1.0	1.0	0.7	1.0	1.0	1.0	2.0	2.0	0.7	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	2.0	1.0	3.0	3.0	3.0	3.0	
Code 23	2.3	2.0	1.0	1.0	1.0	1.0	1.7	1.0	1.7	1.0	1.0	2.0	1.0	1.0	1.7	1.0	1.0	1.0	1.3	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	3.0	3.0	3.0	3.0	
Code 24	2.0	2.0	1.0	1.0	0.7	0.7	1.0	1.7	1.3	2.0	1.0	1.3	0.7	1.0	2.0	1.0	0.7	1.0	2.0	2.0	2.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	3.0	3.0	3.0	3.0	
Average	2.3	2.3	2.0	1.3	1.3	1.1	1.5	1.8	1.9	1.8	1.3	1.5	1.2	1.4	2.2	2.0	1.1	1.4	2.1	2.0	2.0	1.0	1.0	1.0	1.0	1.0	1.0	3.0	3.0	3.0	3.0		
AQMC	3.0	2.0	1.0	1.0	1.0	1.0	1.0	2.0	3.0	1.0	1.0	1.0	1.0	1.0	3.0	3.0	1.0	1.0	2.0	3.0	1.0	2.0	1.0	2.0	2.0	3.0	3.0	1.0	3.0	3.0	3.0	3.0	

(continued on next page)

	8a	9a	9b	10a	10b	10c	10d	11a	11b	12a	12b	12c	12d	12e	
Code 1	3.0	1.0	2.3	2.3	2.3	2.0	1.0	2.3	2.3	2.3	2.0	2.0	2.0	2.0	103.2
Code 2	3.0	2.3	2.3	2.7	2.0	2.0	1.0	2.0	2.0	1.0	1.3	1.3	1.3	1.0	95.6
Code 3	2.0	1.0	2.0	2.3	2.3	2.0	1.0	2.0	2.0	2.0	2.0	2.0	2.0	2.0	92.2
Code 4	3.0	1.7	2.0	2.0	2.0	1.7	1.0	2.3	2.3	2.0	2.0	2.0	2.0	1.0	91.7
Code 5	3.0	2.0	2.0	2.0	2.3	1.0	1.0	2.3	2.3	2.0	1.3	1.3	1.3	1.3	89.0
Code 6	3.0	2.0	2.0	2.0	2.3	1.0	1.0	2.3	2.3	2.0	1.3	1.3	1.3	1.3	88.7
Code 7	2.0	2.0	2.0	2.0	2.0	2.0	1.0	2.3	2.3	2.0	2.0	1.3	1.3	1.0	87.3
Code 8	3.0	2.0	2.0	2.0	2.3	2.0	1.0	2.0	2.0	2.0	1.3	1.3	1.3	1.3	87.0
Code 9	2.0	2.0	2.0	2.3	2.0	1.0	1.0	2.3	2.3	2.0	1.0	1.3	1.3	2.0	83.3
Code 10	3.0	1.0	1.0	2.0	2.0	2.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	79.1
Code 11	2.0	1.0	1.0	1.0	2.0	1.0	1.0	2.0	2.0	2.0	1.0	1.7	1.0	1.7	76.0
Code 12	2.0	1.0	1.0	1.0	2.0	1.0	1.0	2.0	2.0	2.0	1.3	1.3	1.3	1.3	75.4
Code 13	2.0	0.7	2.0	1.0	2.0	2.3	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	75.4
Code 14	2.0	1.0	2.0	2.0	2.0	2.0	1.0	1.7	1.0	1.0	1.7	1.7	2.0	1.7	74.9
Code 15	2.0	1.0	1.7	1.7	2.0	1.0	1.0	2.0	2.0	2.0	1.0	1.0	1.0	1.0	72.7
Code 16	2.0	1.0	2.0	1.0	2.3	1.0	1.0	1.0	1.0	2.0	1.0	2.0	1.0	1.0	72.6
Code 17	2.0	1.0	1.0	1.3	2.0	1.0	1.0	1.0	1.0	2.0	1.7	1.7	1.3	1.7	71.8
Code 18	2.0	1.0	1.0	1.0	2.0	1.0	1.0	2.3	2.3	2.0	1.3	1.3	1.3	1.3	69.8
Code 19	2.0	1.0	1.0	1.7	2.0	2.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	67.4
Code 20	2.0	0.7	0.7	1.0	2.0	2.0	1.0	2.0	2.0	1.0	1.0	2.0	1.0	1.0	67.3
Code 21	2.0	1.0	1.0	1.0	2.0	1.0	1.0	1.0	1.0	2.0	1.0	1.7	1.0	1.7	66.4
Code 22	2.0	1.0	0.7	1.0	2.0	2.0	1.0	2.0	2.0	2.0	1.0	2.0	1.0	1.0	64.8
Code 23	2.0	1.0	1.0	1.0	2.0	1.0	1.0	1.0	1.0	2.0	1.0	1.7	1.0	1.7	64.1
Code 24	2.0	0.7	1.0	0.7	2.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	62.5
Average	2.4	1.3	1.6	1.7	2.2	1.6	1.0	1.8	1.8	1.8	1.4	1.6	1.3	1.4	81.7
AQMC	2.0	1.0	1.0	3.0	3.0	3.0	1.0	3.0	3.0	2.0	1.0	2.0	1.0	1.0	87.0

## **Exit Question Responses**

### **Positives (What is working in your environment.)**

This section contains raw, unedited input from the assessment team members.

- Shared tools (reuse and commonality). The tools are not optimal and they cause some problems, but they still lend value and help the process.
- Framework concept is working well. Provides a structure and clarity in terms of process.
- Code teams are cooperating (not competing).
- Small team peer level communications works well, due to small teams, co-location, knowing each other. (The challenge is providing objective evidence as needed).
- Small team (about 3).
- Requirements and design (including prototype) process – expect significant paybacks in later phases.
- Developing an architecture for software.
- Distributed team is working due to good communications (emails, phone/conference calls, video conference, ...).
- SIERRA Tools offloads a lot of work and provides version based code defense.
- SIERRA Tools handles platform (build) issues (compilers, file transfer, I/O issues).
- SIERRA Tools provides LXR.
- Communications (email, phone calls).
- Open (flat) environment that promotes a highly productive team.
- Team members who have worked with each other for some time.
- Team structure (co-located developers).
- Team buy-in to the vision (for the code).
- Emotional commitment to the goal (building a culture).
- Just enough process, just enough tools philosophy targeted at the “easily implemented improvements” at a given time (or phase).
- Lots of direct customer contact for multiple team members. Helps with the quality of communications, level of trust, and clarity of requirements, and prioritization.
- Matrix structure provides access to resources and skills.
- Flat teams (little formal hierarchy).
- Personnel are open to discussion and criticism.
- Team is using a modified extreme Programming approach, it is working very well. Pair programming works well for complex issues, and when mentoring of new team members is needed.
- The team is using a release checklist that identifies each task and who is responsible for the task. Each member is expected to take ownership (flat, self managed team) ownership of at least one task.
- Shared successes within a team.
- No single point of failure (task ownership and skills) within a team.
- Analysts and developers on the same team (actually usually the same people).

- Co-located and under the same management so issues/conflicts easily discussed and resolved.
- IP driven focus so the teams are lean and mean.
- Direct communications (co-located and know each other).
- Maintaining serial and parallel code versions.
- Maintaining dual codes (new and legacy versions). (this is also identified in the risk section).
- Informal communications.
- Excellent people.
- Organizations and people want to do the right thing.
- Small, co-located teams.
- Good relationships with CSU (infrastructure issue).
- Using issue tracking to make sure team is working on the correct issues, and nothing else.
- Independent (with in team) QA verification and review role is working very well.
- High performing, focused team with a collective team vision that is cooperative, supportive, and provides team help, e.g., by mentoring.
- Collective problem solving.
- Flexibility.
- Simple peer pressure enough to keep people from checking in defective code.
- CVS.
- Communication, e-mail, meetings both formal and informal, telecommunication, video conferencing, the Web, etc.
- Providing information via the Web, e.g., Designs, Theory Manuals, User Manuals, Coding Standards, Bug Lists, Issues, "Living Documents", etc.
- An academic research and development atmosphere.
- Developers are analysts as well and of similar background.
- Good initial design which has been for the most part followed.
- Use of the inline documentation tool Doxygen, coding standards, etc.
- Good algorithms people – state of the art.
- Meeting notes which are posted on the Web either directly or in eNotes or possibly via other packages.
- Automatic testing at check-in or nightly regression testing and subsequent notification.
- Collocation of team which can include experimentalists as well.
- Streamlined process, fast prototyping.
- Customer interaction, customer/user on team.
- SIERRA Framework/Tools – commonality, confidence, nightly testing, library support.
- Distribution of work, that is, the work can done in parallel.
- Good leadership and support from management above, good task planning.
- "Good unwritten rules" which are usually followed.
- Small team.
- Strong believers in SQE.
- A non prescriptive choice of tools, e.g., Merant Dimensions (both pro and con), DOORS, BugZilla, eNotes, Gnats, Rational, MS Project, GANNT

- charts, PURIFY, PURECOVERAGE, CVS, AltaVista, Mantis, APROBE, FORESYS, Flint, OnYourMarkPro, TogetherSoft, SourceForge™.
- Developer Sandbox and Junkyard.
  - Checklists and sign up sheets for tasks.
  - Paired programming with no heroes and the sharing of successes and responsibilities.
  - “Just enough process, just enough tools”
  - Co-location of code team (7).
  - Communication (7).
  - Small code-team (6).
  - Nightly regression testing (4).
  - Team rapport (3).
  - Manager attends code team meetings (2).
  - CVS repository (2).
  - Respect and good interaction between developers and customers (2).
  - SIERRA Tools and support (2).
  - Checklist for release process.
  - Underlying philosophy of writing and releasing code.
  - Just enough tools, just enough process.
  - Management support.
  - Team believes in common vision – emotional connection.
  - Modified extreme programming.
  - Checklist of requirements open to all for assignment and completion – all team members can see progress on requirements.
  - Pair programming – avoid stuck points, no points of failure or glory
  - Regular code team meetings.
  - Sandbox for separating algorithm development from code.
  - E-notes and Mantis work well for tracking team meetings.
  - Interest in getting job done.
  - High regard for correct answer
  - Rapport with systems people.
  - Algorithm writers.
  - Having analysts and code developers the same individual. Less likely to work outside the plan.
  - Can use each other’s code.
  - Formal structure for documenting code.
  - Figured out how to work remotely.
  - Code teams working together.
  - Clarity of tasks.
  - Taking the time to design and working toward a flexible design.
  - Tasks divided up thoughtfully.
  - Competent Team – high performance.
  - Development manual and Users guide on the web.
  - Flat organizational structure provides everyone with a voice in decisions, debate and criticism.
  - A good mix of technologies.
  - Management support for all types of work not just research.

## Barriers

This section contains raw, unedited input from the assessment team members.

- Not all groups subject to the same structure (structure seems to mean process and SQE).
- Tools are a barrier (existing tools need to work better).
- We need to be worrying about the following, but there does not seem to be any analyst community demand (pull) for it.
- Solution verification.
- Algorithm verification.
- Acceptance criteria is difficult to get from customers/analysts since they are still getting used to providing requirements.
- Some tools (mentioned PVCS Dimensions) are not working – too difficult to use – payback too low.
- Complexity of SIERRA Framework is a barrier. (it is also asset due to its complexity).
- Tools turn around is too long (specifically SIERRA Tools).
- Too many non-productive requirements.
- A more efficient implementation is critical for tools and frameworks; but there will be a learning curve once done. (SIERRA)
- Documentation is not done or out of date for tools and framework (SIERRA).
- SQE tools and support (stated as a resource issue related to issue tracking, requirements tracking, test execution, ...).
- SIERRA Framework – not well documented, a lot of “tribal knowledge”.
- Testing is a problem when using the SIERRA Tools and Framework. Framework S/W breaks. (Feels that the SIERRA Architect group does not always do adequate testing of their own).
- Not all teams are thinking forward to coupled Mechanics codes.
- Some teams are using regression testing as their safety net (inadequate developer testing) as their first line of defense.
- Interdependencies between teams.
- Loosing work time due to
  - o Network loads.
  - o CPU availability (need more access and more processors).
  - o SIERRA check-in.
  - o SIERRA Tools inflexibility – has been a problem – still is a problem – is getting better.
- Time constraints (Schedule pressure and resources).
- Resources, especially testing resources.
- Tools and support. Very little is available yet the push seems to be toward tools!
- Need more test skills and test resources (funding and skilled resource issue).
- Funding mechanisms spread the \$'s too broadly and prevent development of the full spectrum for electrical codes.
- ASCI funding priorities (not funding some things that should be funded today).
- Matrix Structure – people may have the right skills, but there is still a steep learning curve related to team approaches. Also the learning curve for a complex code can be lengthy.

- Test bed does not support adequate granularity for the test suite (only supports a narrow range of tests).
- Management through milestones does not work if the milestones are arbitrary or trivial. Milestones should be subject to quality and value criteria, otherwise the motivation is to create low quality (poor) milestone targets.
- ASCI Apps – DOE interface.
- Highly distributed team (not co-located, not even all in the same state) requires more oversight (but can still work well).
- Marketing of codes to ASCI SNL Apps and DOE.
- Lack of SQE background.
- Lack of SQE tools.
- Time away from primary jobs (like the assessment).
- Lack of SQE resources.
- Lack of funding for SQE activities.
- Lack of SQE training and direction.
- Lack of time (time, time, time!!!).
- Any SQE process that does not add value (no process for process sake).
- Proscriptive process that makes it too difficult to release developer (or beta, or field test, or ...) versions.
- Communications as it relates to documented process.
- Tools for traceability.
- Lack of available staff (bodies and skills).
- Succession planning (new staff to replace older staff – time to mentor, learning curve, ...).
- Continuing changes in compilers, O/S's, debuggers, TPS tools, SQE tools takes time away from the primary job.
- Small teams have to do a lot themselves.
- Access to computing resources (ASCI White) – this is critical because of the milestones.
- Scope changes, requirements creep, constantly changing priorities.
- Lack of direct access to customers and their requirements (second hand requirements).
- Having budget but not the skilled people to fill the positions.
- Resource allocation.
- Lack of an Infrastructure, e.g., in the area of tool support, testing, and investigation, documentation, Software Engineering, Code Testing and its requisite setup, etc. Thus there are questions relative to matrixing and PMFs .
- An over prescriptive process can shift away from development.
- Research more important than programming, i.e., the clash between research and production.
- System administration and the need to do someone else's job.
- Uncontrolled changes in operating systems, compilers, libraries, etc. and other types of related dependencies. In addition, there are ASCI Program dependencies and code (analysis) dependencies.
- Members in various locations with possibly limited travel budgets.
- Ignorance of the generic nature of the IP, milestones for milestone's sake not capability.

- Budget cuts, prioritization, scope creep, and the bureaucracy - "Schedule can be job one".
- Insufficient time and reward to do documentation, overlooked.
- Insufficient process for traceability, e.g., from user requirements, where, when, why to levels above.
- Over prescriptive documentation of a fluid process, e.g., documentation of meetings.
- Not enough verification money - the bulk of what there is has been allocated to validation.
- A lack of understanding of the time consuming nature of the SQE process.
- Integration of validation data into code (SQA wise) and although not part of this Assessment, how to validate a code.
- Issue tracking.
- SIERRA Framework/Tools not well documented, inefficient, cumbersome, and complex. Verification tests need to be accommodated by the nightly regression tests.
- Lack of examples for placing a code under SIERRA.
- Number of computer processors has not grown with the increased load on SIERRA.
- "Traceability of who broke the system".
- The believe that the regression tests are a cure all for problems.
- Need a longer Beta test period.
- Machine accessibility, e.g., Janus.
- Meetings too long.
- No full-time people.
- Not having the same type of organization structure for electrical simulations activities as there are for mechanical and fluid simulation activities, e.g., the electrical group is spread amongst several organizations whereas the fluids/mechanical is in one organization.
- High learning curve.
- Testing not granular enough.
- An insufficient test bed, i.e., lacks a SIERRA like test bed.
- Marketing at Sandia.
- Lack of a window into other projects (APPs) milestones.
- Lack of tool support and a skill-appropriate position to perform SQE for the code team. Belief is that code team should be working in their areas of expertise and not being diverted to learn new tools for SQE (4).
- Do not have the man-power for SQE function (4).
- Documentation is not rewarded or time allocated – need manager support (3).
- Dimensions tool (3).
- Plenty of money, but not plenty of qualified staff (2).
- Code team does not have time to evaluate SQE tools (2).
- Changing customer requirements and funding (2).
- No full-time FTE's assigned to project so it is easy to have code team members pulled to work on other projects (2).
- SQE is diverting resources from work (2).
- Physical distance between code team members (2).

- Schedules do not incorporate code team dependencies (2).
- Meaningless IP milestones. The level is too high and report progress by the number of milestones completed does not provide an accurate picture.
- Software is not modularized.
- Steep learning curve before contributions to project can be made.
- Unable to put unit tests into testbed.
- Funding mechanism, coordination and support does not address technical issues the developers see as very important issues for the future.
- Marketing software within the ASCI environment.
- SQE process documentation, especially for small code teams, does not seem to add value.
- Co-location hurts SQE documentation.
- Team meetings are too long.
- Change to compilers and operating system upon which the code team depends
- Improve automated process for regression testing.
- Spending too much time with the tools
- A systems person needs to check out code for a code team – but other code teams check-out their own code.
- On-again off-again direction.
- Time to document practices is not there.
- Some lack of flexibility of SIERRA Tools.
- Too much reliance on regression testing only.
- Tests were broken due to incomplete testing of the framework.
- Testing thinking is too local and not global.
- SIERRA Framework is not documented well enough.
- SIERRA Framework is abstract, complex, and a barrier to all code teams.
- Tools are cumbersome.
- Traceability – code teams have tools by the importance of traceability has not been acknowledged.
- Need more verification money.
- Stability of priorities.
- Need automated testing.
- Clash of cultures – research, publishing, academia vs. SQE.

### **The Opportunities (Expected outputs of the assessment)**

This section contains raw, unedited input from the assessment team members.

- Help code teams understand what will be required of them in the future.
- Identify/fix GAPS.
- Improve practices document – provide clarity.
- Management challenge – find communality to fix issues like:
  - Email archive.
  - Other infrastructure.
- Generic process recommendations that can be shared (best practices, best implementations, more shared infrastructure, best tools).
- Improve the practices document – it does not add value in all areas.
- Teams want feedback and direction.
- Pressure to management to provide SQE resources (e.g. testing process).

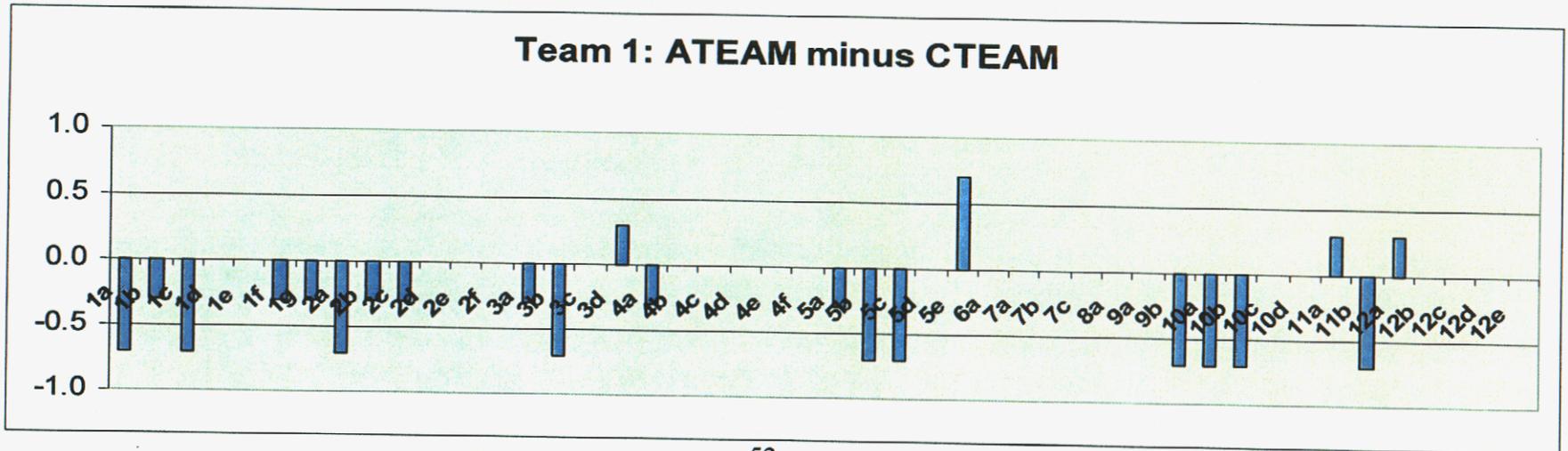
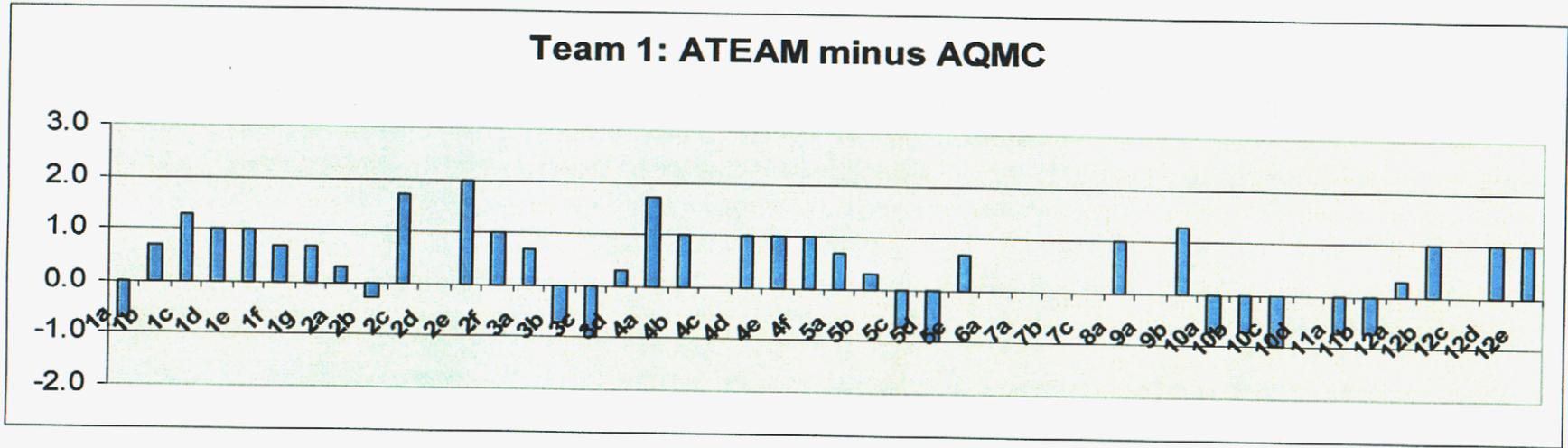
- Records management – need direction from ASCI.
- Present strengths/weaknesses (GAP).
- Improvement in scheduling and tracking of teams.
- Find out if Doors is the Requirements Management tool of choice.
- Need a better organizational level understanding of responsibilities between developers, analysts, can customers. Especially in the area of requirements and acceptance criteria.
- Tangible positives for the teams (funding for tools, understanding how the assessment contributes to the ASCI quality needs).
- Team understanding why the assessment is valuable.
- Identified toolsets that the teams do not have to manage (tools infrastructure and support). Support is critical! Input into the tool selection is critical!
- Examples of best (better) practices.
- Any useful checklists.
- Do not be proscriptive, offer options.
- Identify GAPS.
- Tools.
- Structure.
- Criticism.
- Help on schedule/resource issues.
- Derive practices/processes that can be drawn upon by others.
- Identify common flaws and solutions.
- Clear direction.
- Evaluation of schedules (are we doing enough fast enough).
- \$'s for SQE and QA work.
- GAP analysis (how are we doing).
- Easily implemented improvements (what can be easily do better).
- Provide reasonable targets.
- Guidance, suggestions, not just scores.
- Teams should get a more complete report too (not the all team report, but a more detailed report/directions per team).
- Mentoring for best practices, how to steps.
- Quantify the “easily implemented improvements” as seen by the A-team, e.g., how can meetings be easily documented, what is evidence, etc., “most bang for the buck”.
- Examples of SQE best (better) practices, guidance, and recommendations, particularly with regard to Infrastructure.
- The GAP analysis and how can the C-Teams improve and what will it take; a candid evaluation.
- What are the priorities and a clear statement of direction.
- What are the benefits to the team.
- Where should we be in a year say.
- Provide a better understanding of how the process is supposed to work.
- Prepare those that will be audited with the appropriate training.
- Be suggestive not prescriptive, however some level of prescription may be necessary.

- Tool suggestions and resource suggestions, i.e., who should we talk to amongst the teams for good ideas.
- Provide feedback information on a team's scores.
- AQMC does its job.
- Education of both developer and customer with regard to their responsibilities; possibly at the ASCI Program level.
- Suggestion – treat SQE practices like Milestones.
- How much time should be devoted to creating software artifacts versus developing code.
- Develop a common solution versus a team specific solution.
- Express the concern over the SQA issue of that some of the software that the codes depend on, e.g., EXODUSII, Sets and Fields (SAF), other SEACAS tools, MPI, Insight (Visualization) to name a few are not part of the Assessment and are not included in the ASCI Applications Table in the Deployment Document
- Provide a list of good practices for selection (not prescription or a little prescription) (7).
- Provide guidance as to the appropriate level of improvement (7).
- Training on SQE practices (6).
- Infrastructure for SQE (3).
- Templates for SQE documentation (3).
- Training on providing objective evidence. (3).
- Deficiencies and strengths (3).
- How are others following SQE practices (2).
- How are we doing under the ASCI plan (2).
- Independent view of how SQE is being implemented (2).
- List of tools and would like input to that list.
- A better way of doing releases.
- A good plan to improve where SQE was weak.
- How does SQE help me do my job better.
- Why the SQE assessment is important to others outside of Sandia.
- Benefits of SQE need to be communicated to avoid box-checking.
- Finding common flaws and common solutions.
- SQE requirements are coming from Sandia, ASCI, Col. Pate – too many.
- Automated tools for SQE.
- Improved scheduling of dependencies between code teams.
- How to get supporting documentation in one place.
- Obtain management support.
- Visible progress in future assessments.

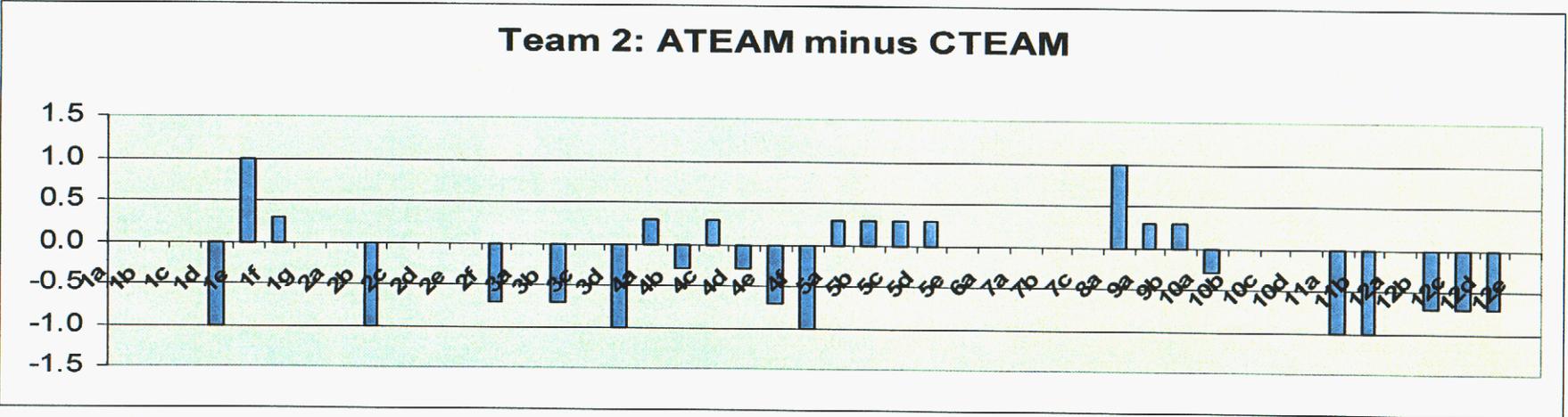
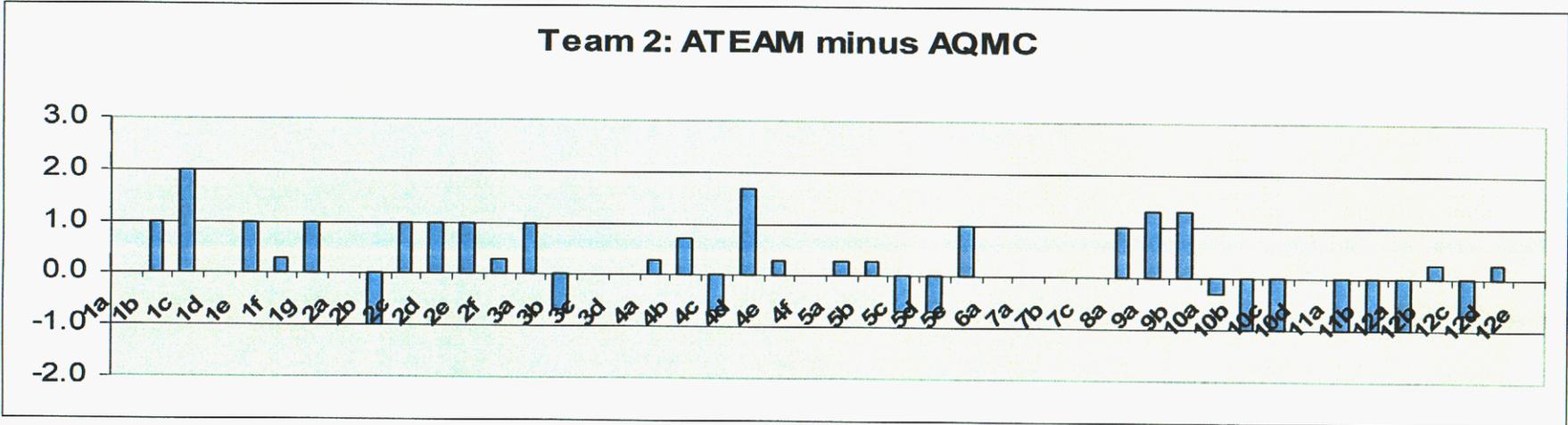
## **Team Plots**

Team 1 .....	53
Team 2 .....	53
Team 2 .....	54
Team 3 .....	55
Team 4 .....	55
Team 5 .....	56
Team 6 .....	57
Team 7 .....	57
Team 8 .....	58
Team 9 .....	60
Team 10 .....	60
Team 11 .....	62
Team 12 .....	62
Team 13 .....	63
Team 14 .....	64
Team 15 .....	65
Team 16 .....	66
Team 17 .....	67
Team 18 .....	68
Team 19 .....	69
Team 20 .....	70
Team 21 .....	71
Team 22 .....	72
Team 23 .....	73
Team 24 .....	74

Team 1

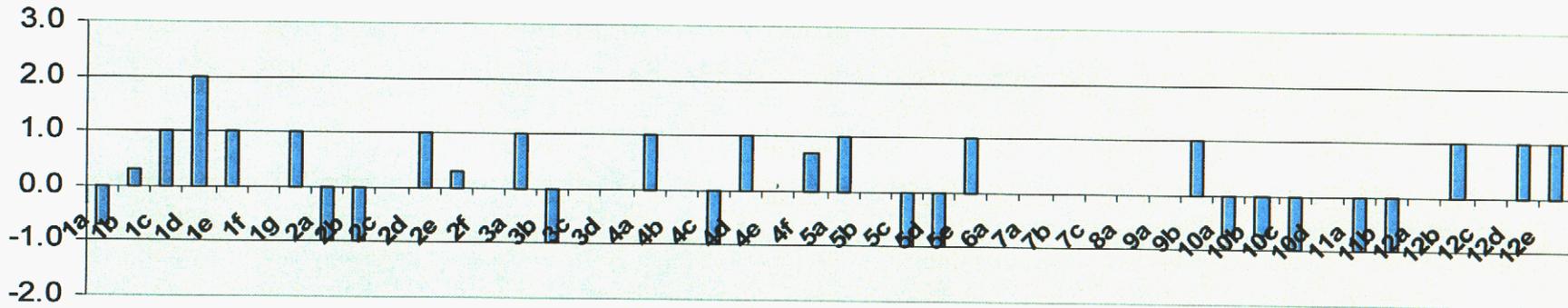


Team 2

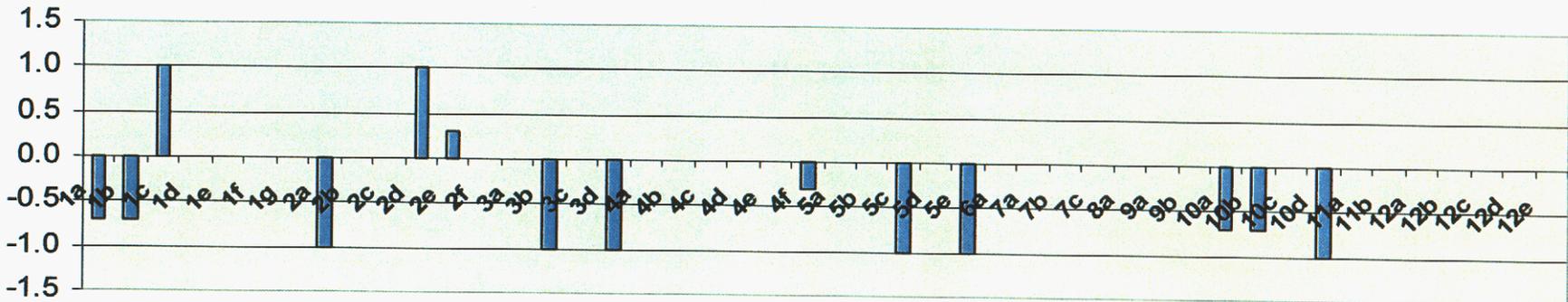


Team 3

Team 3: ATEAM minus AQMC

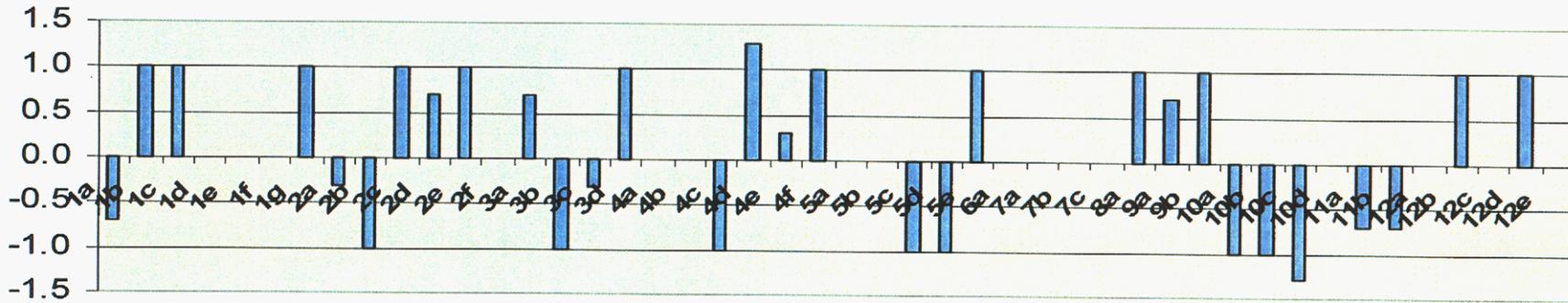


Team 3: ATEAM minus CTEAM

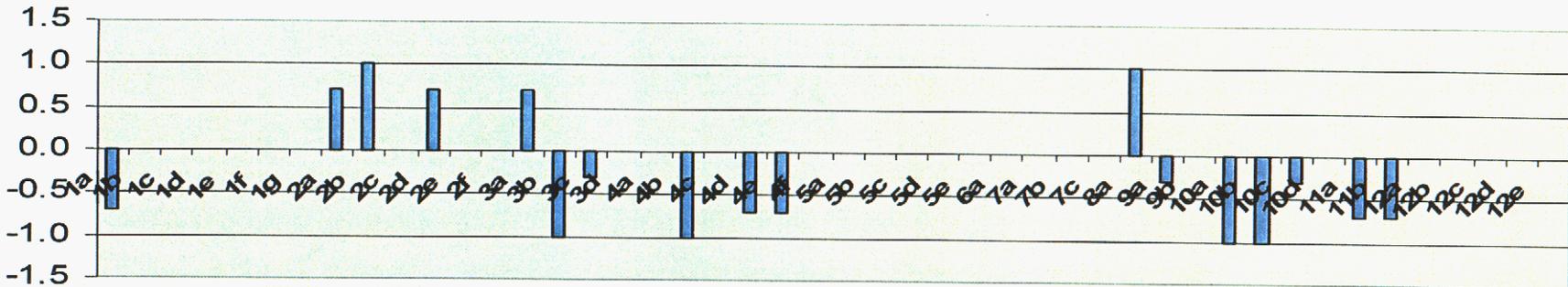


Team 4

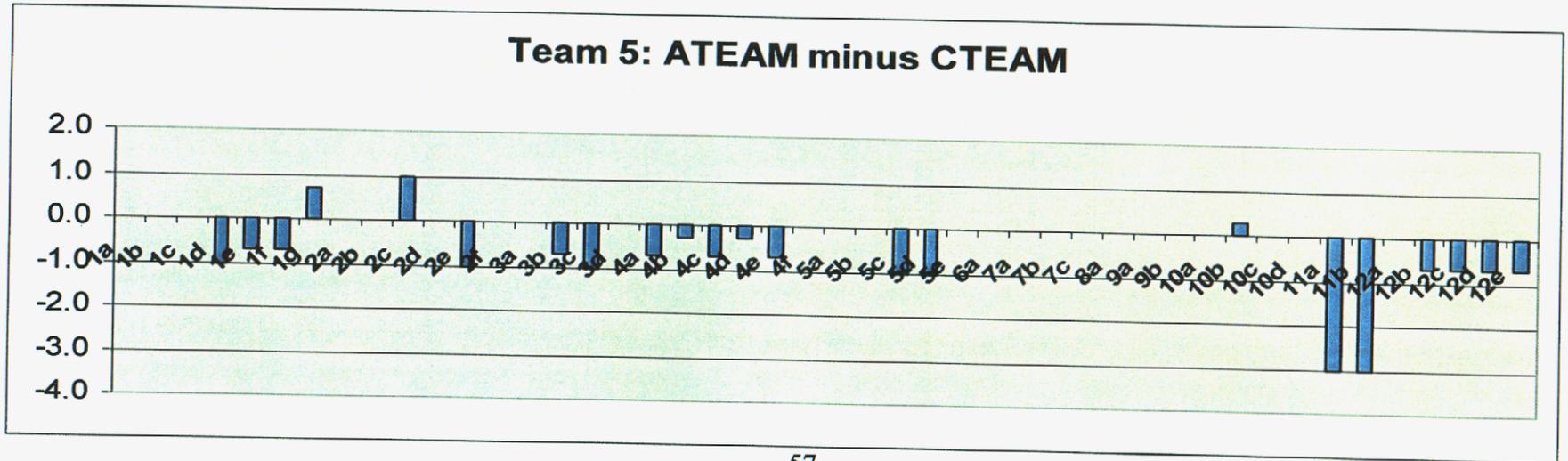
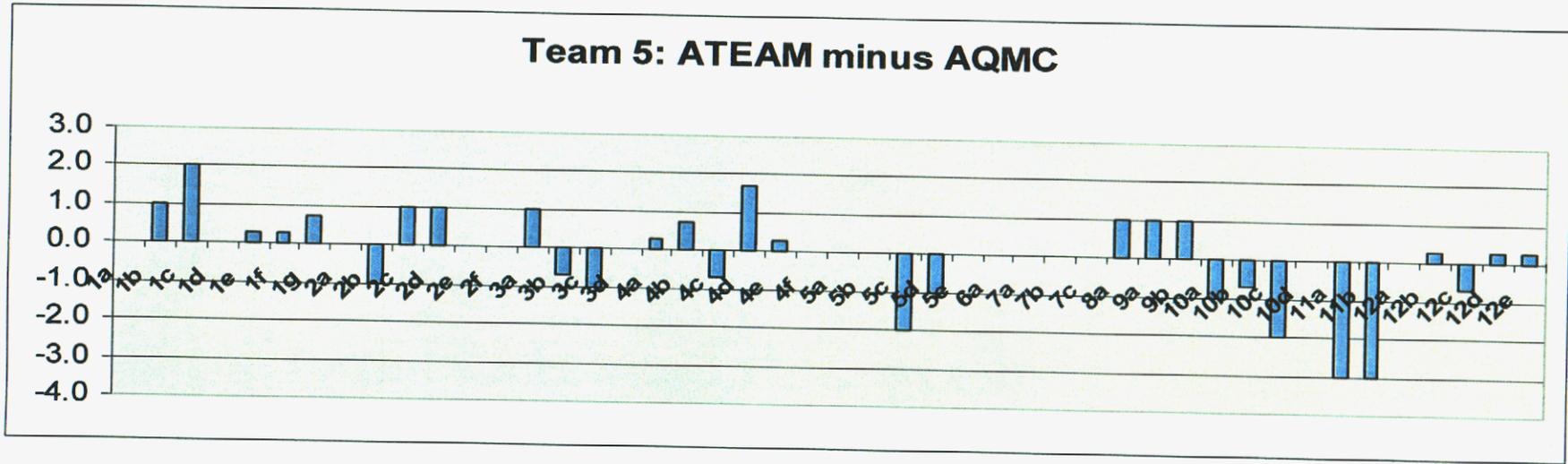
Team 4: ATEAM minus AQMC



Team 4: ATEAM minus CTEAM

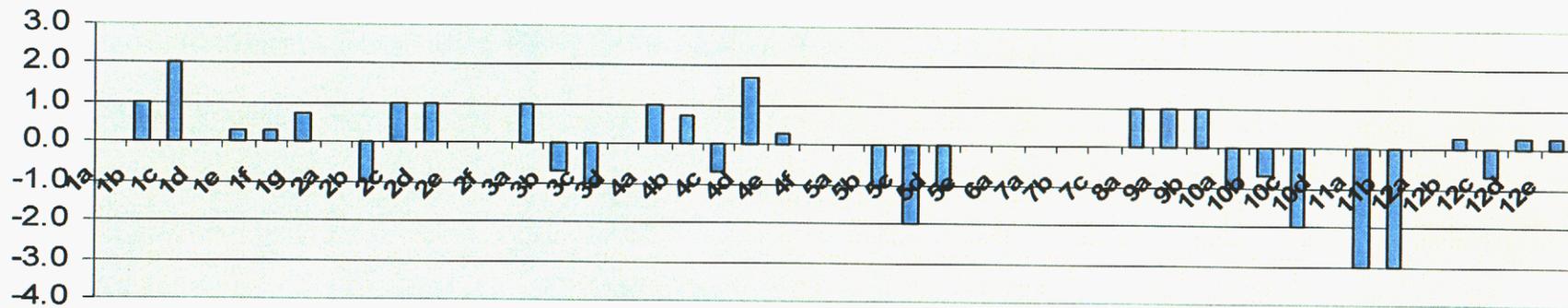


Team 5

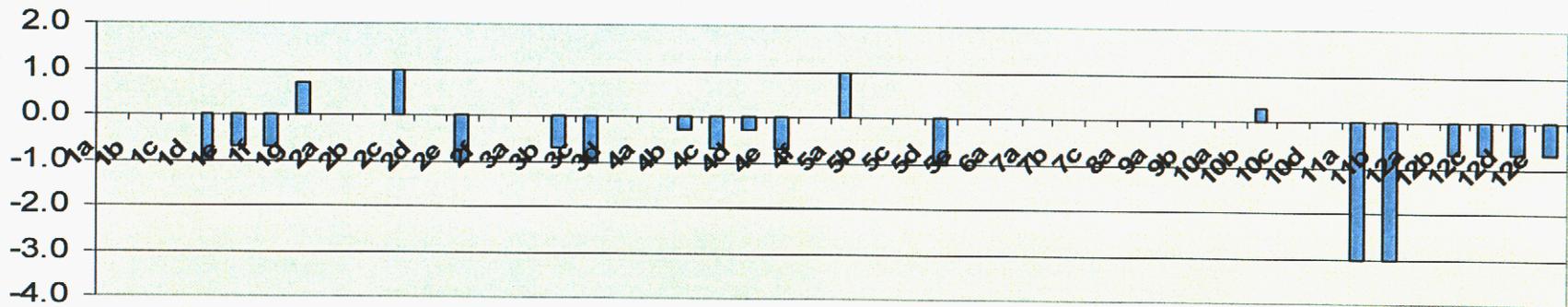


Team 6

**Team 6: ATEAM minus AQMC**

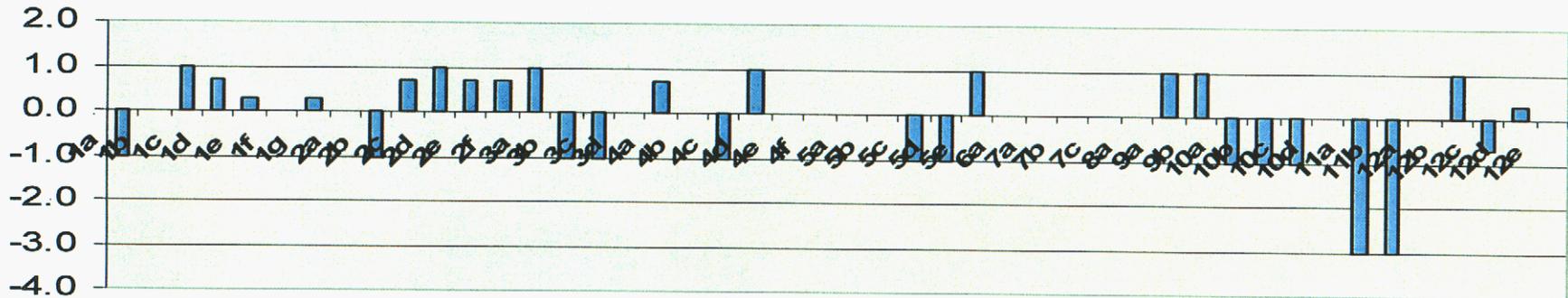


**Team 6: ATEAM minus CTEAM**

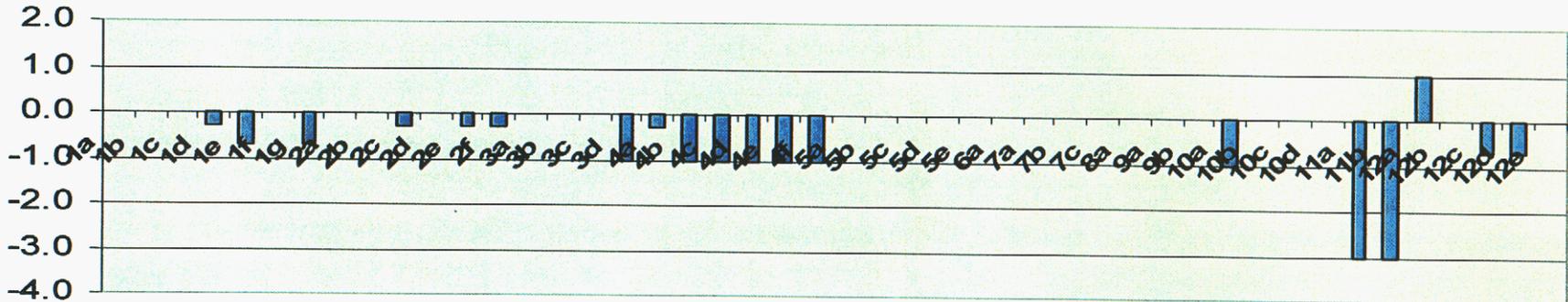


Team 7

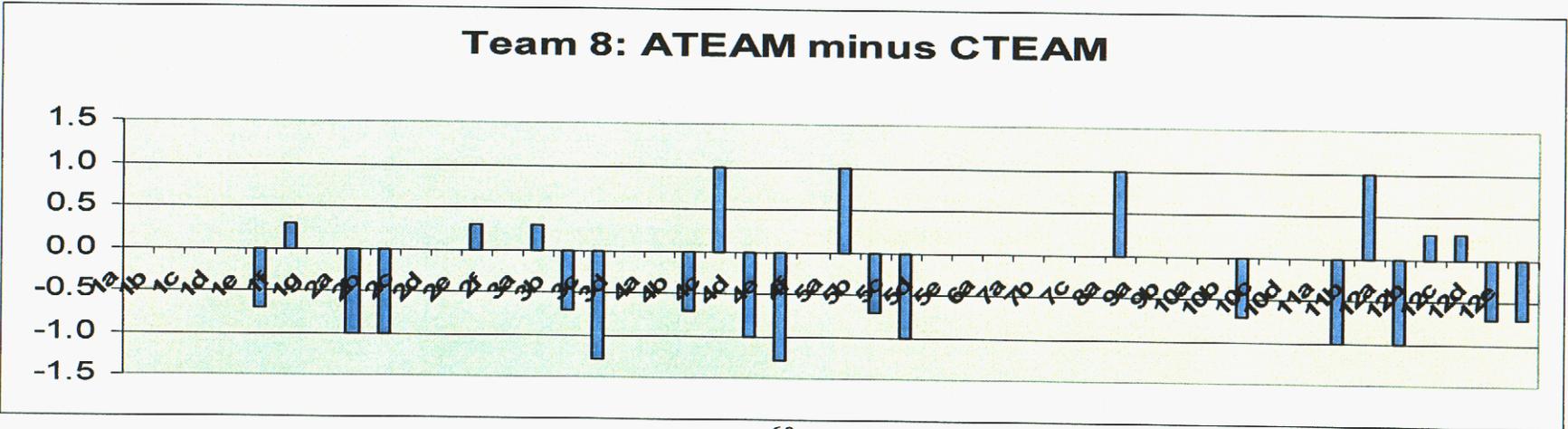
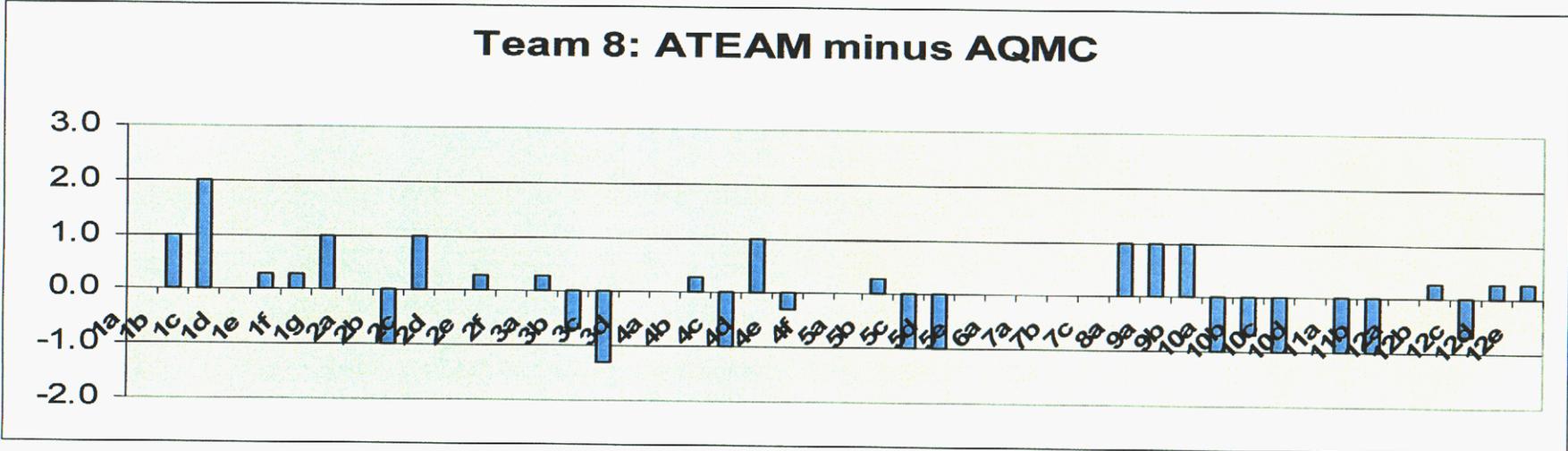
Team 7: ATEAM minus AQMC



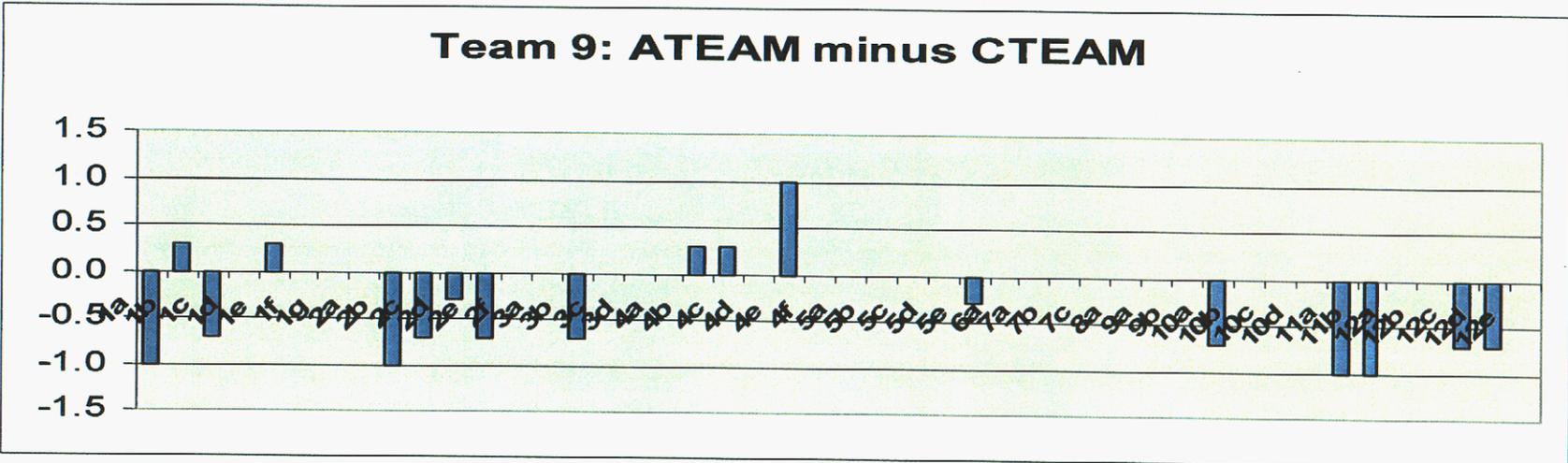
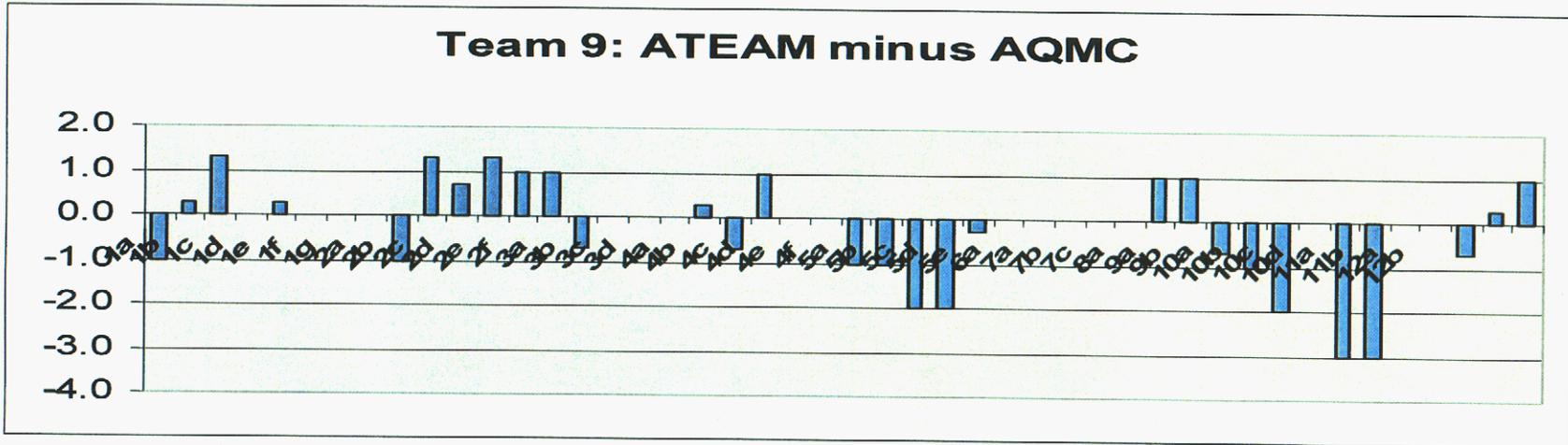
Team 7: ATEAM minus CTEAM



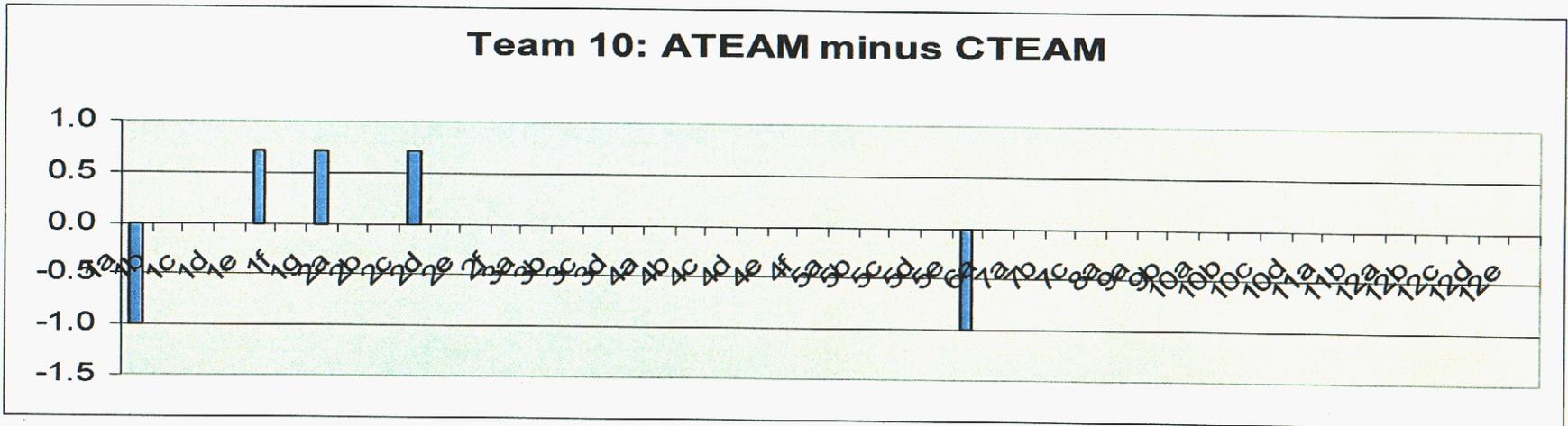
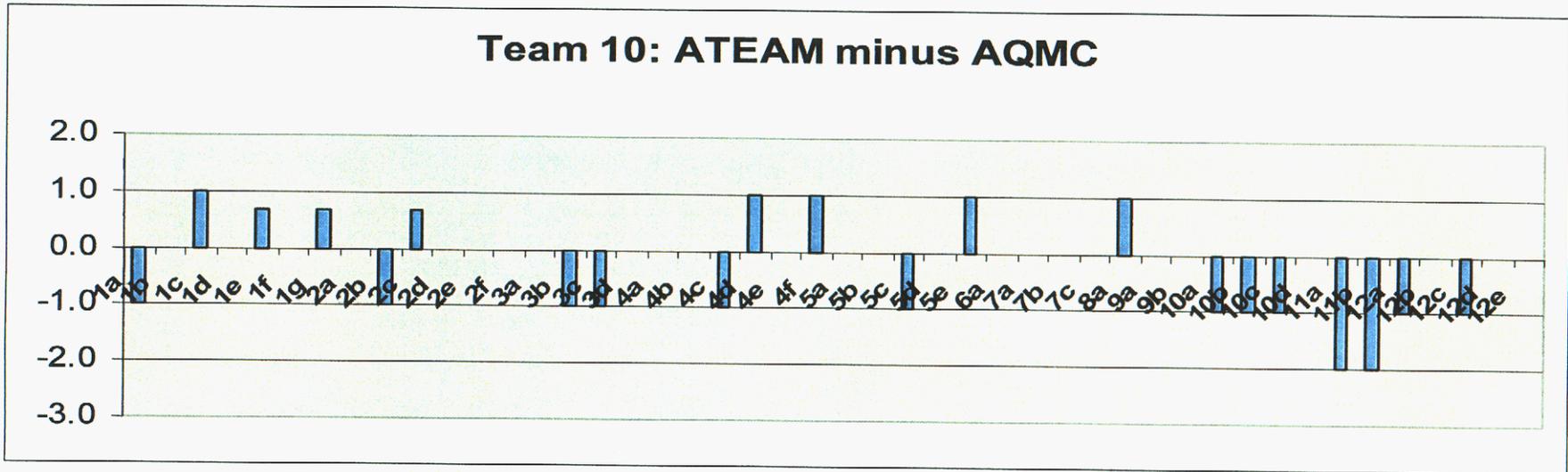
Team 8



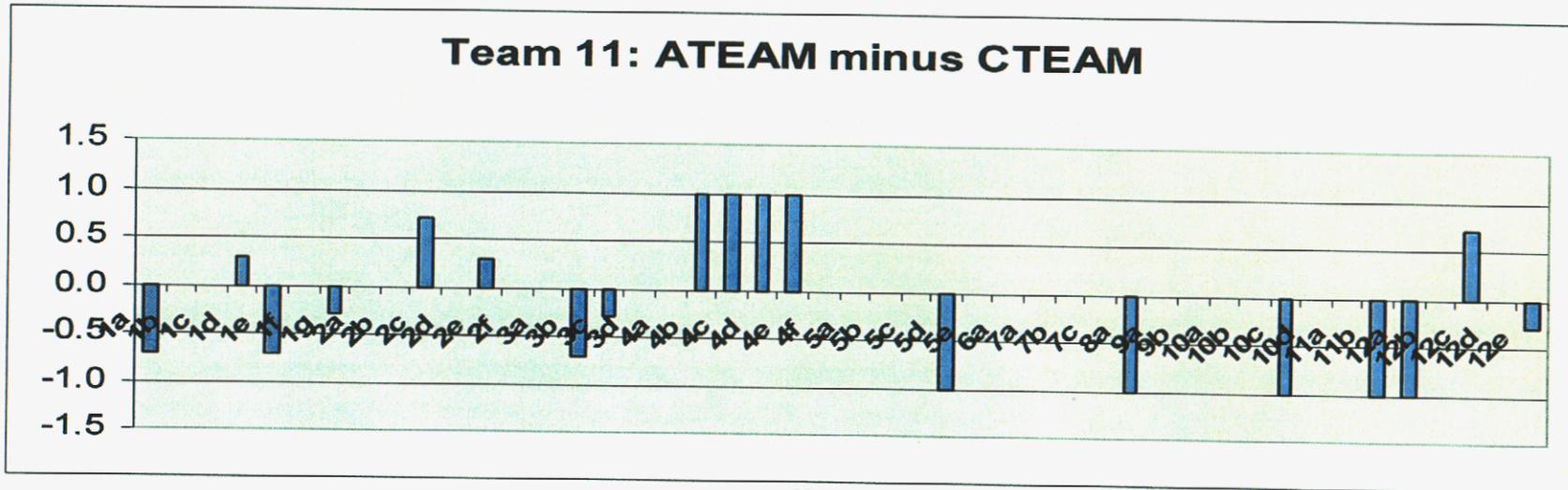
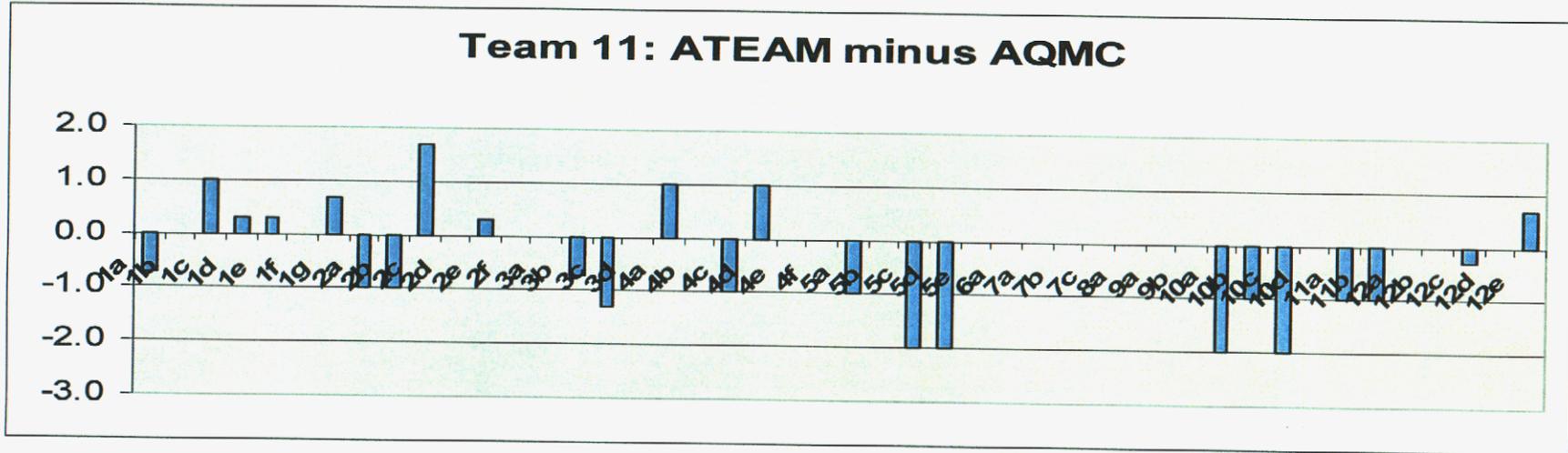
Team 9



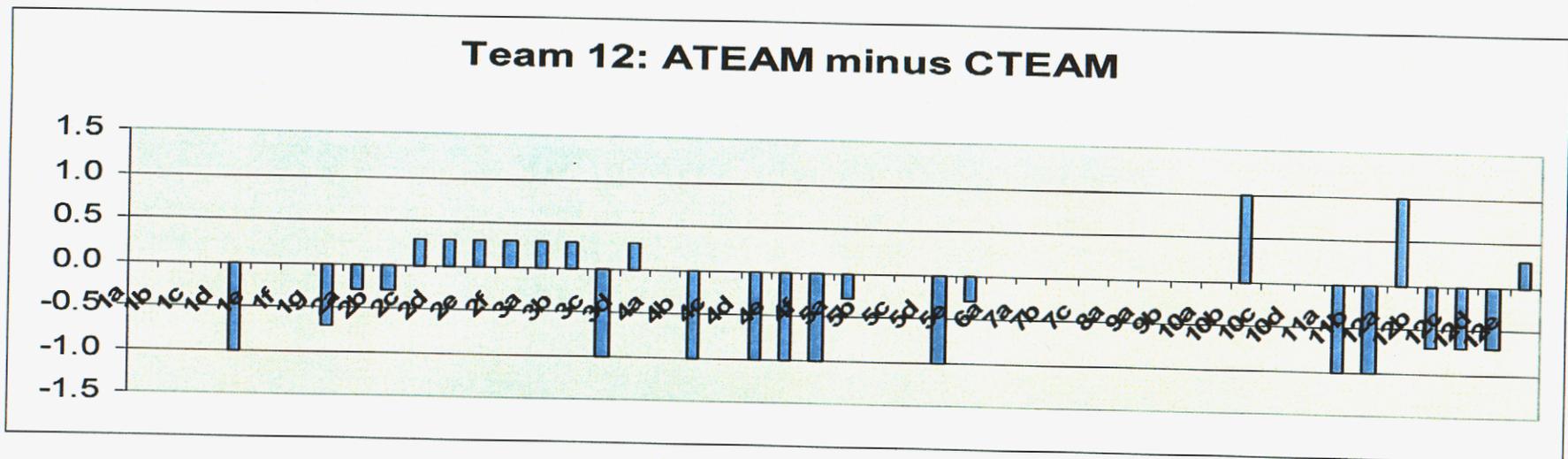
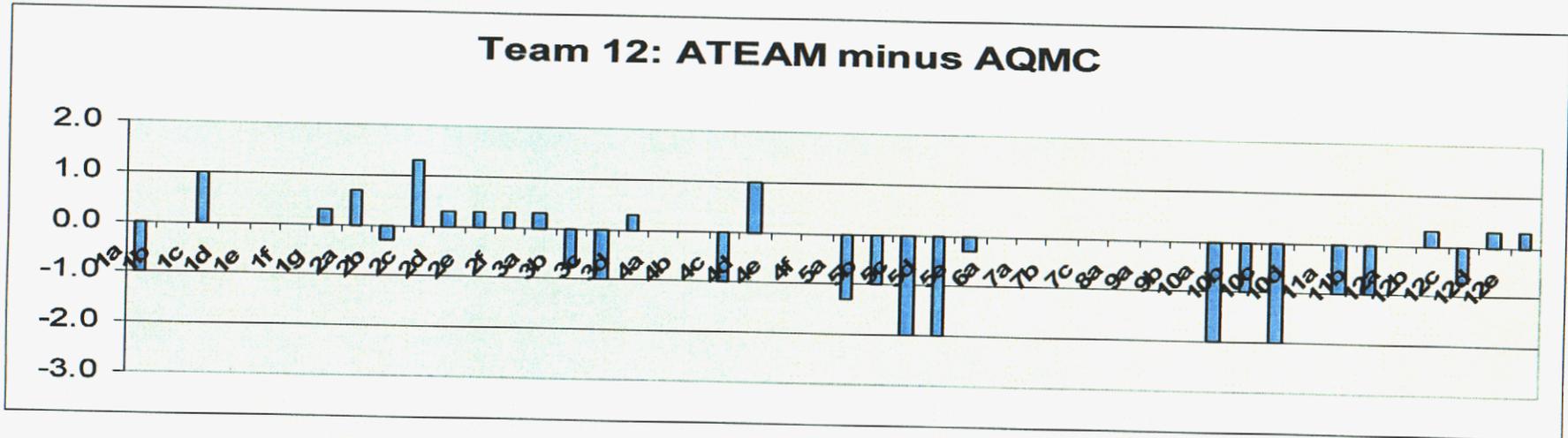
Team 10



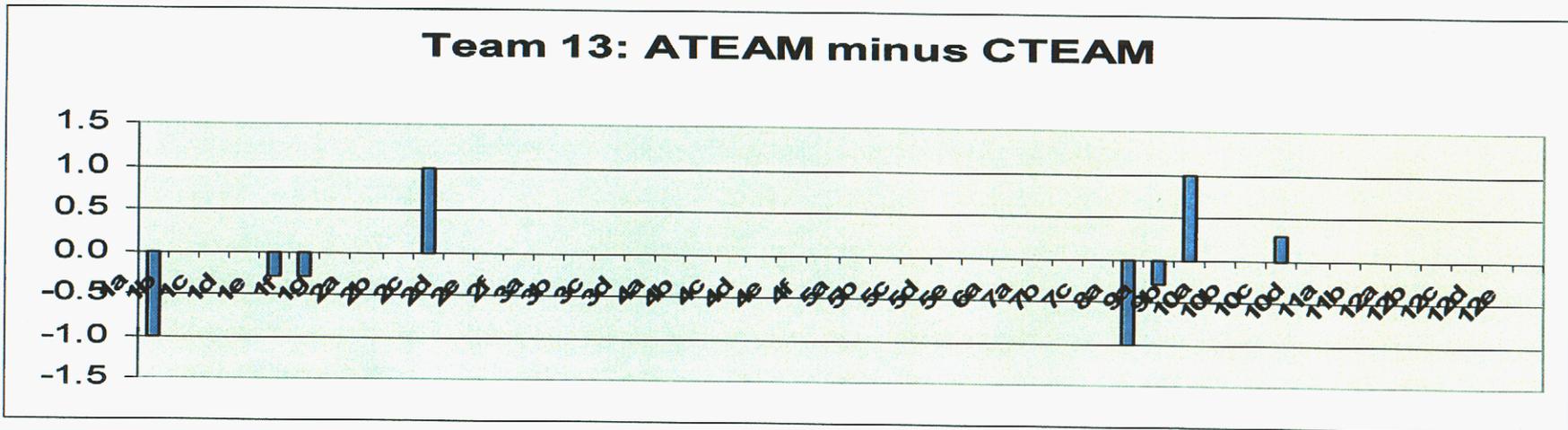
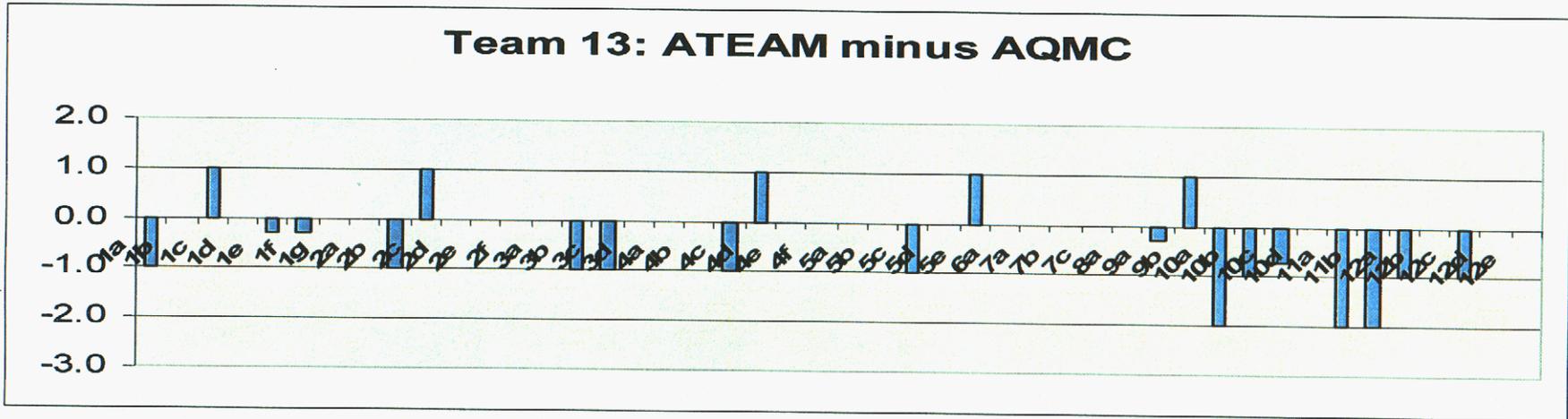
Team 11



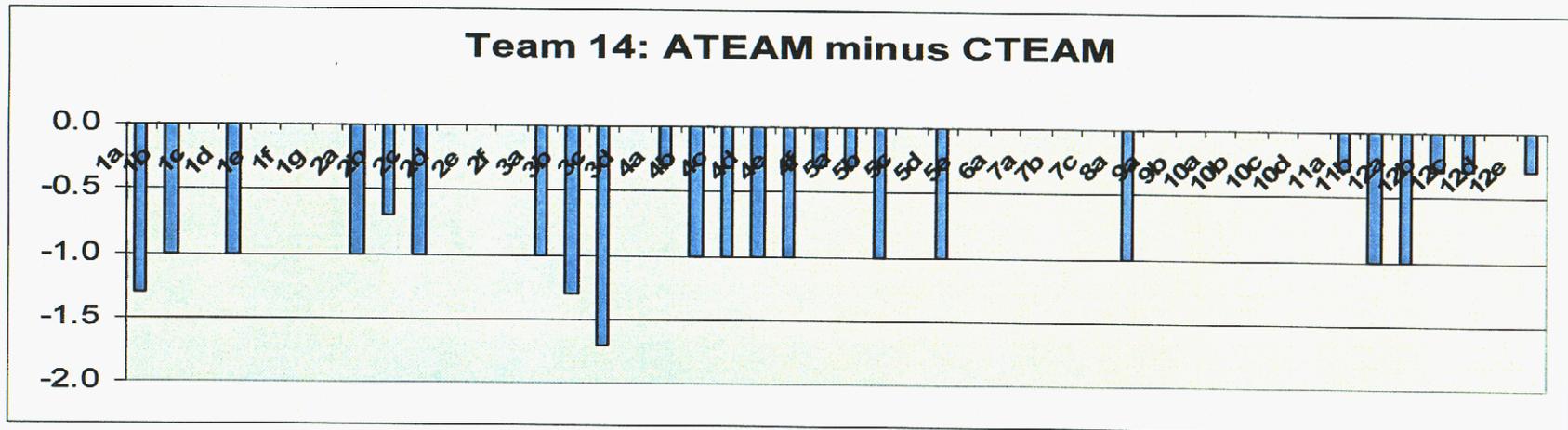
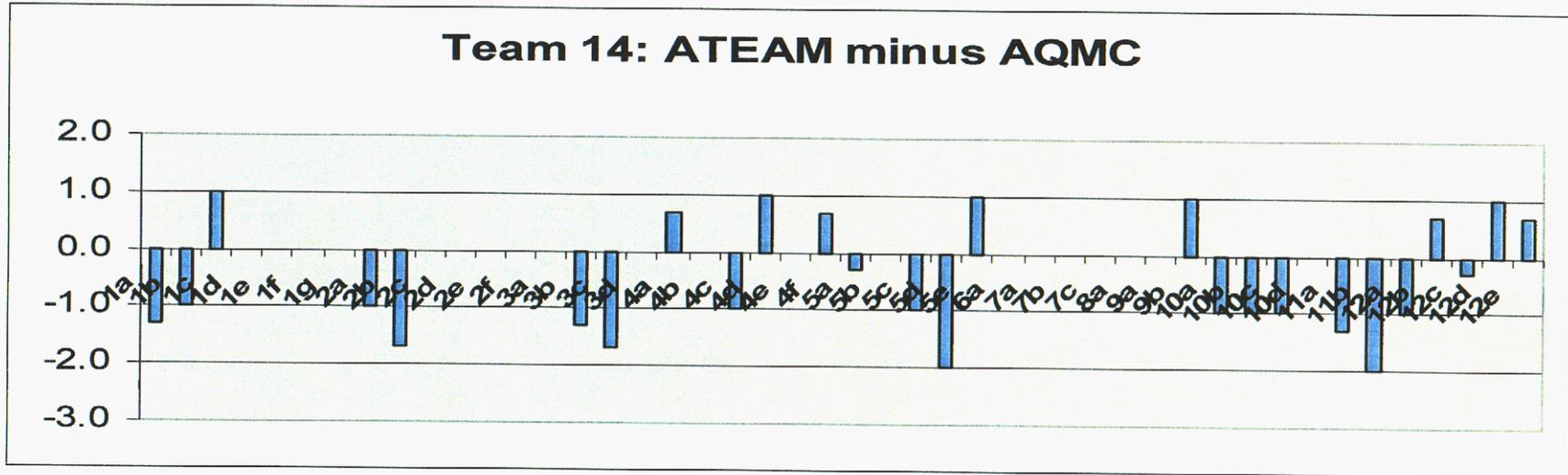
Team 12



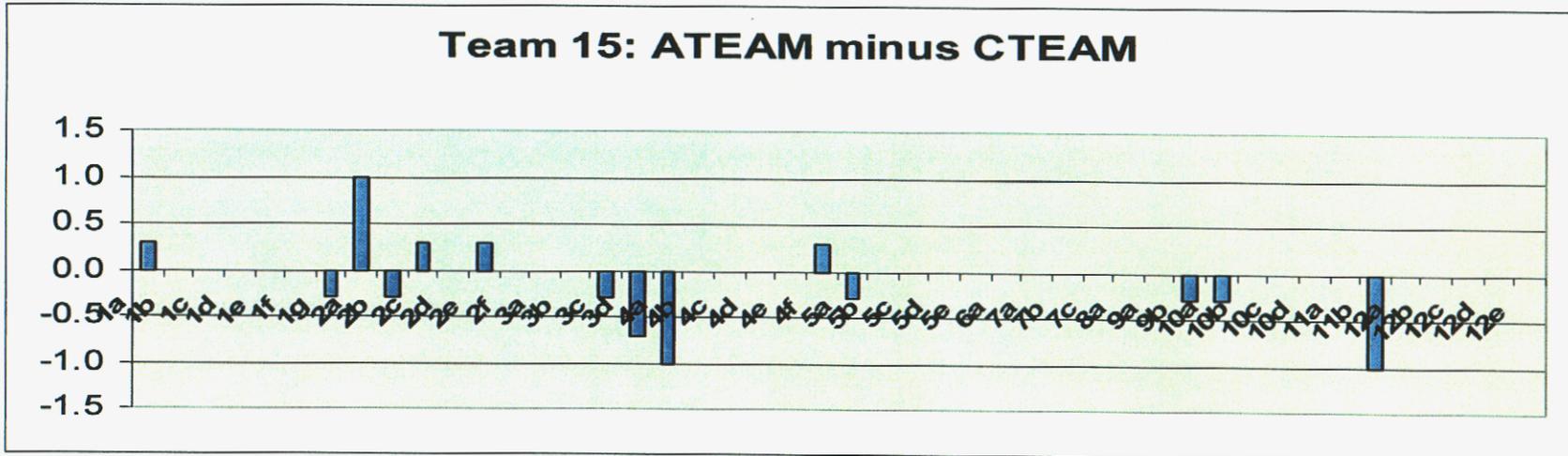
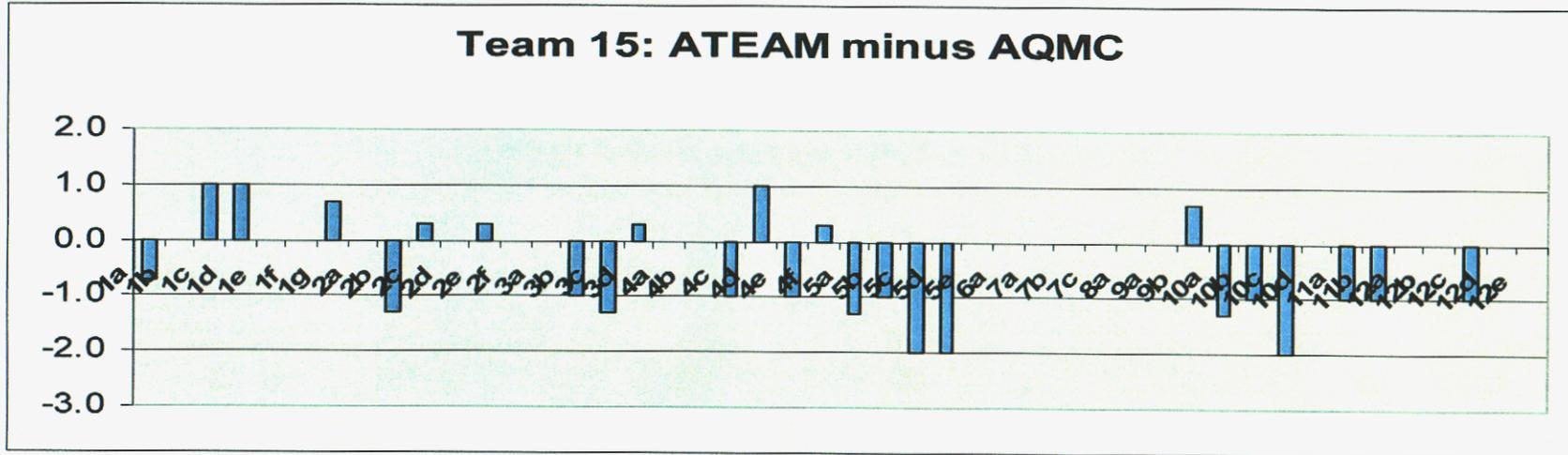
Team 13



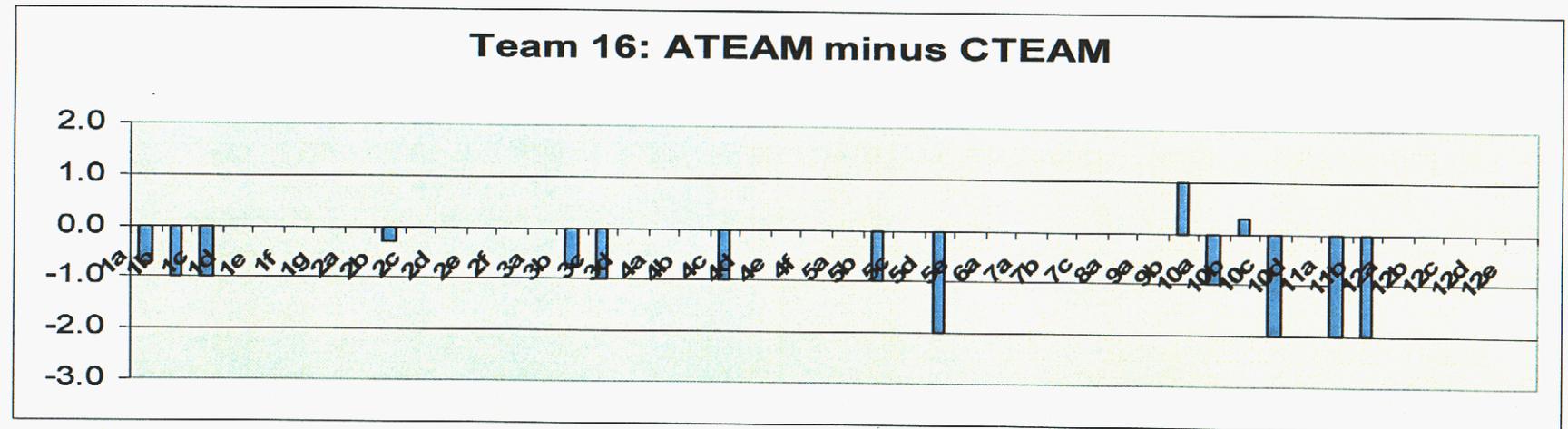
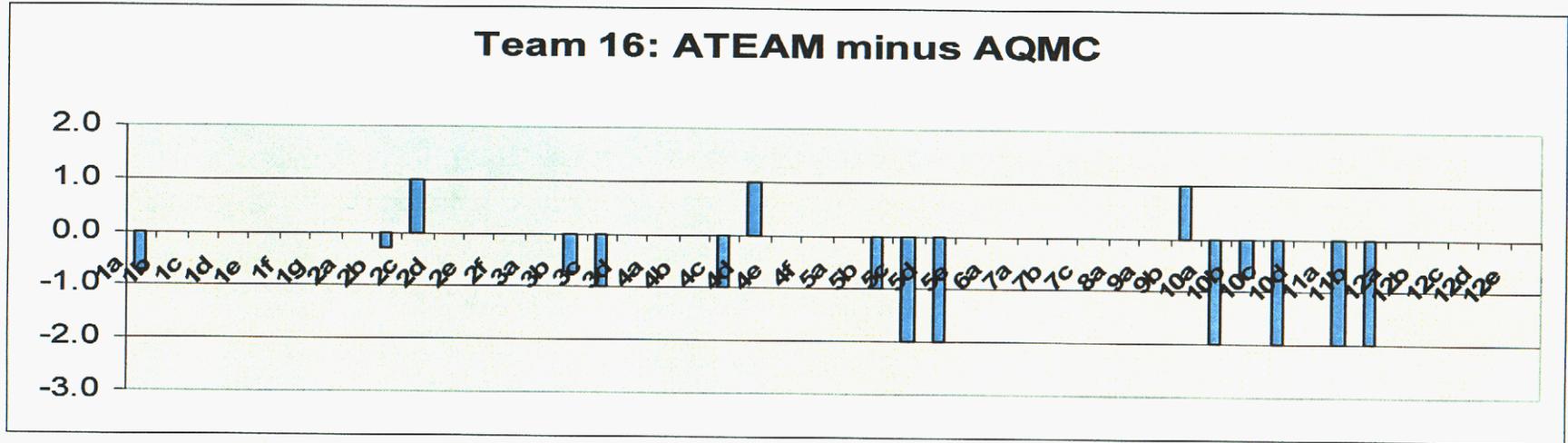
Team 14



Team 15

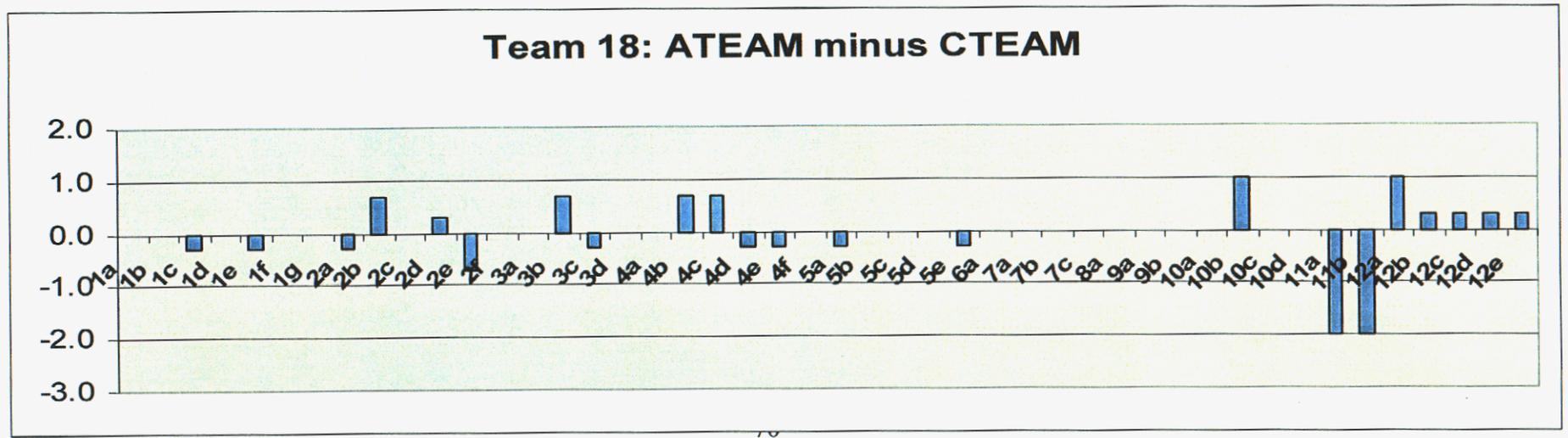
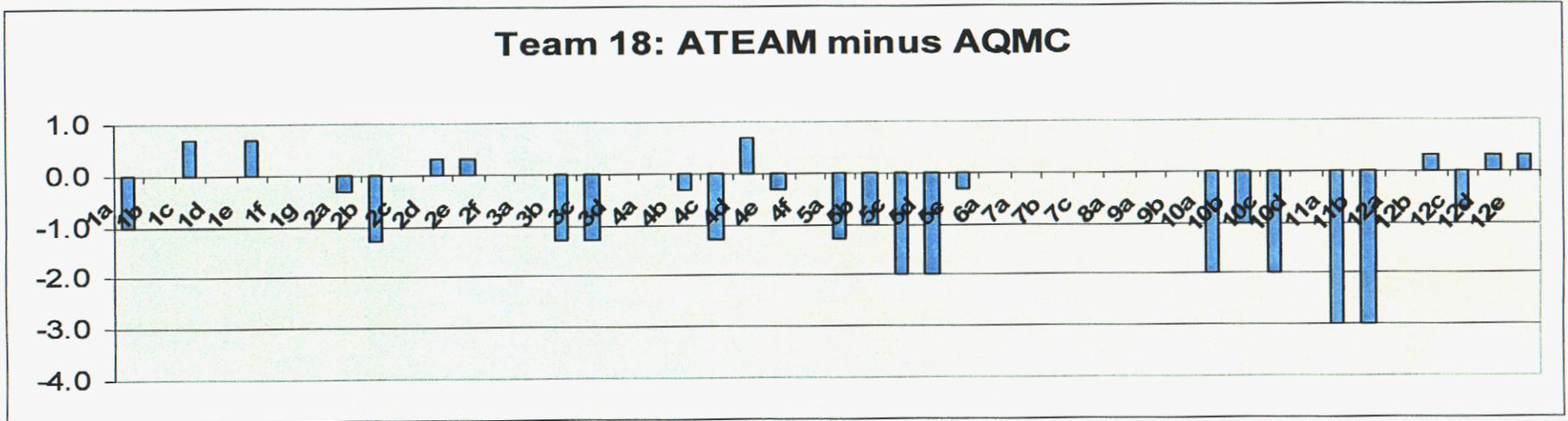


Team 16

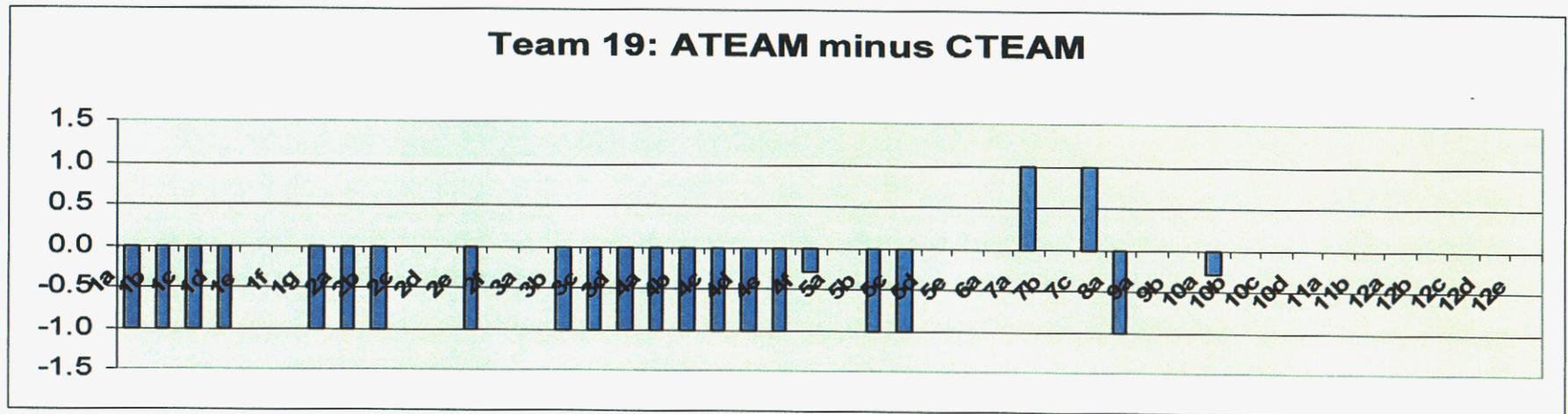
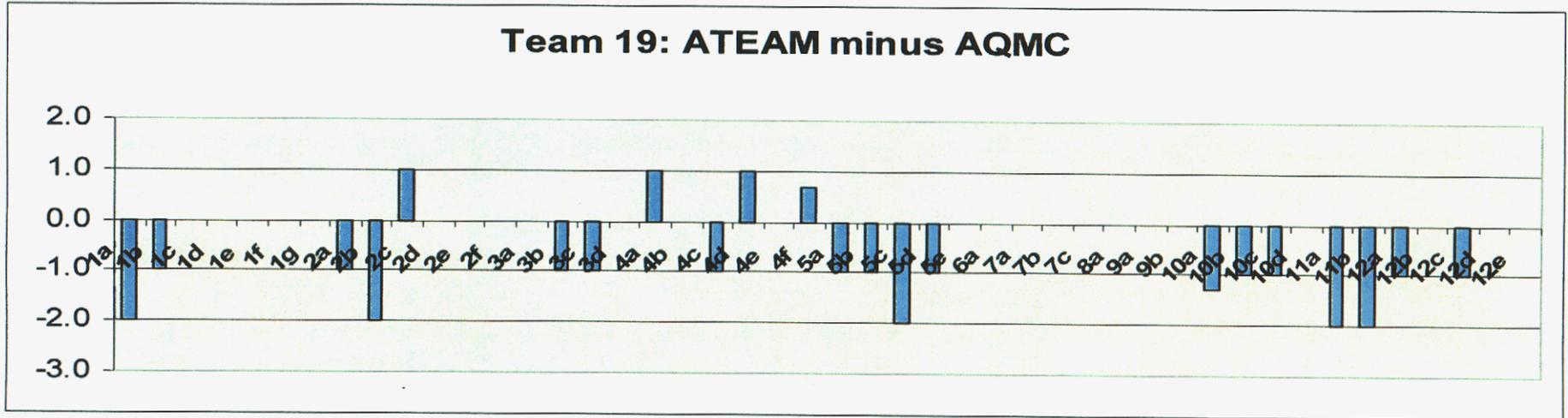




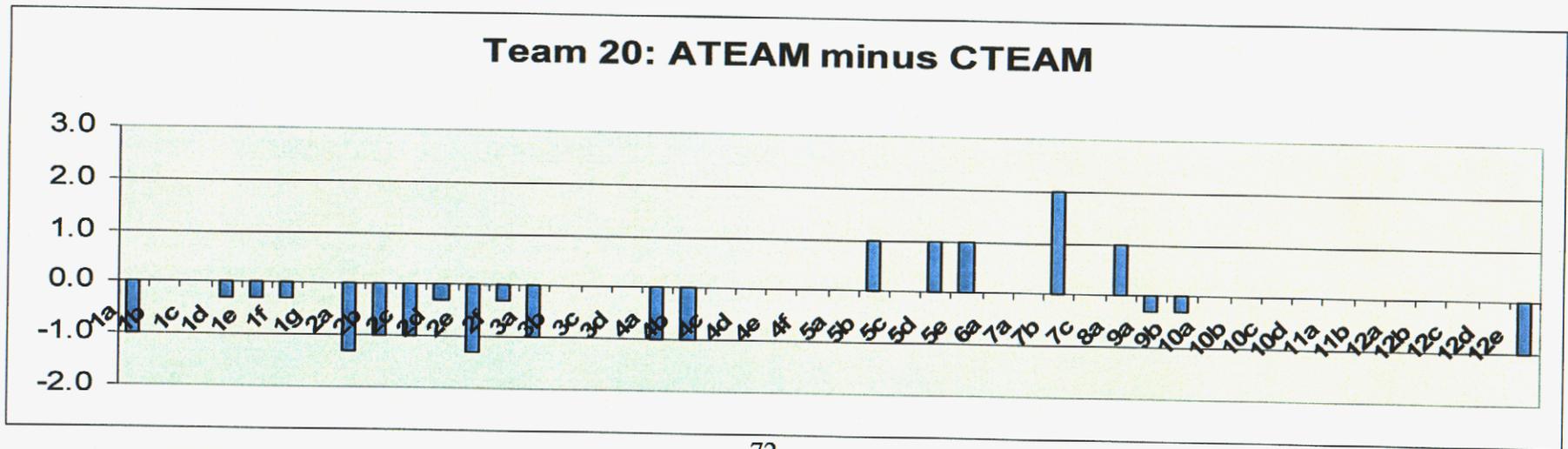
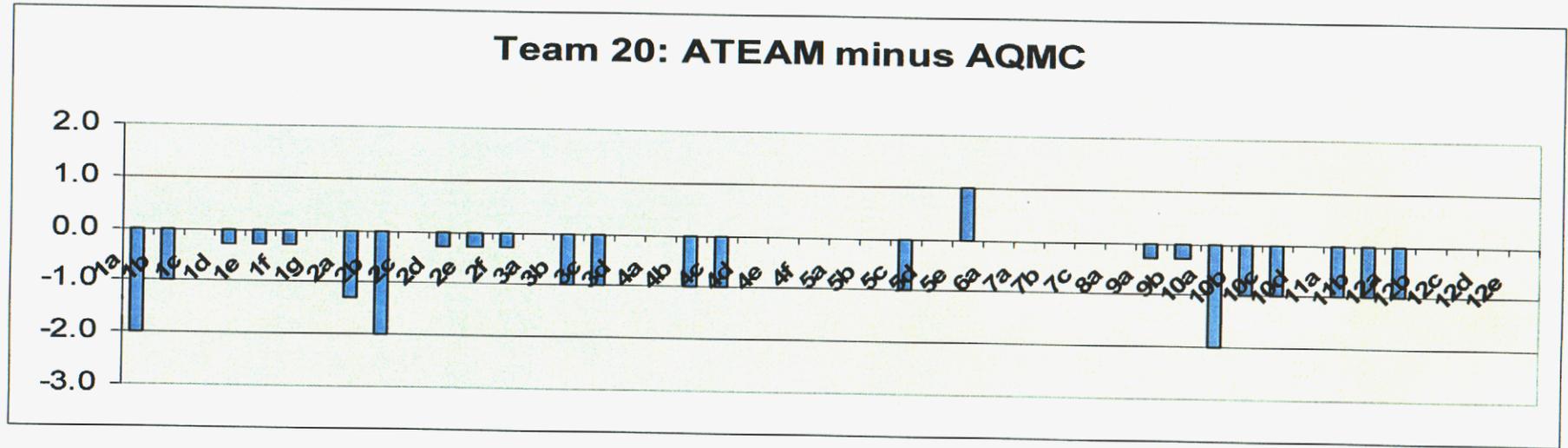
Team 18



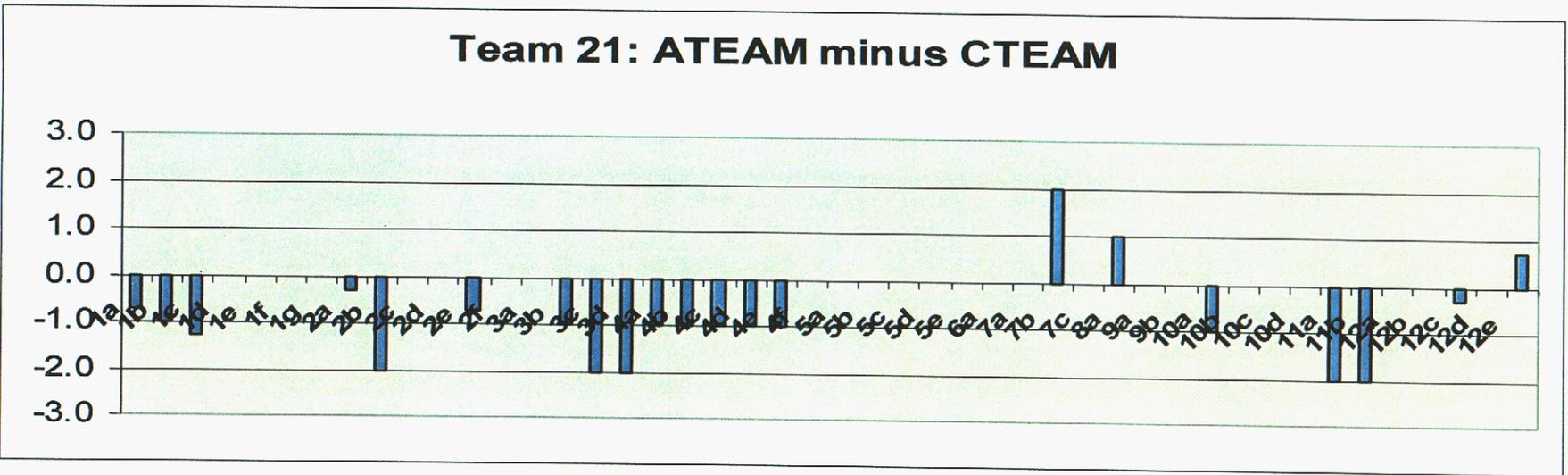
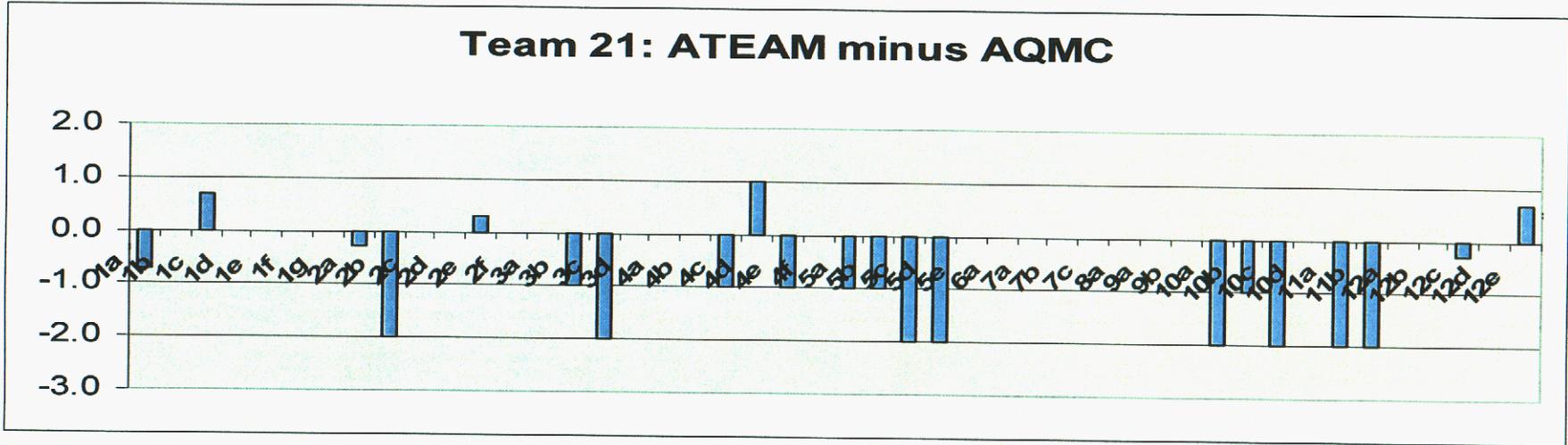
Team 19



Team 20



Team 21

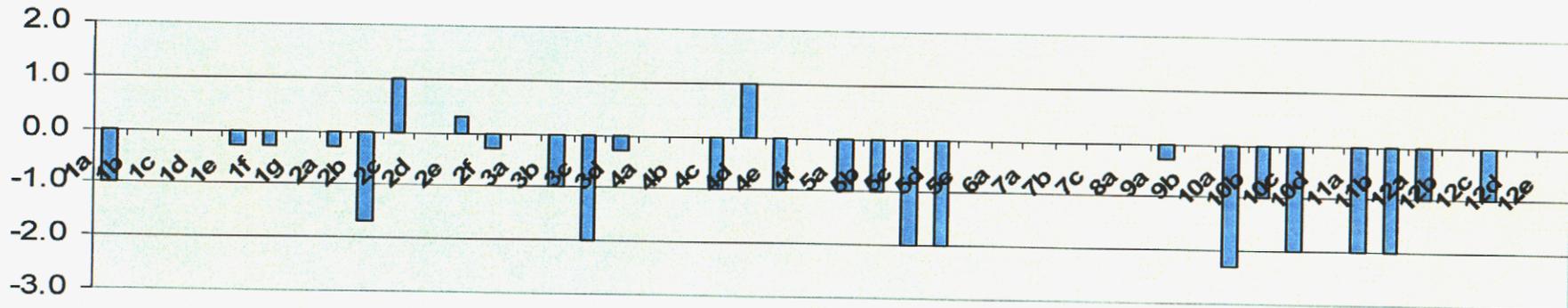




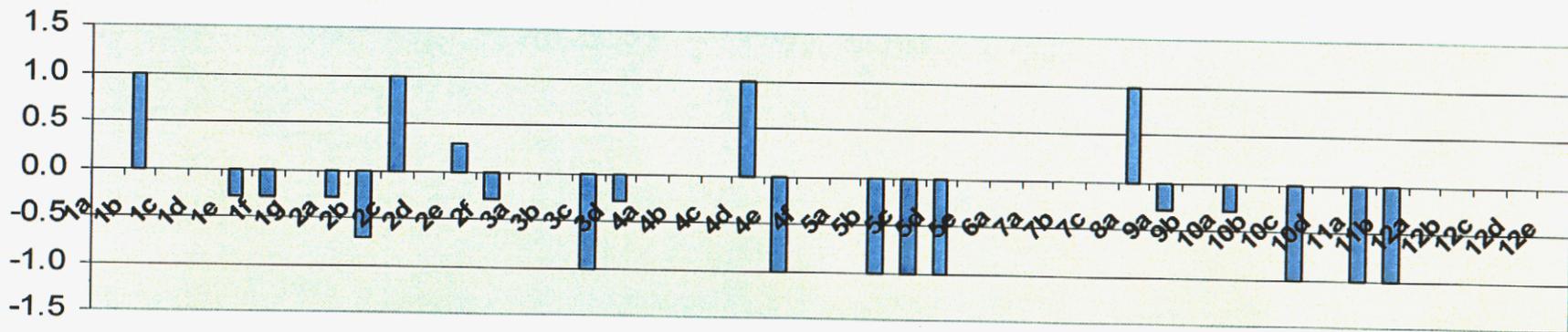


Team 24

Team 24: ATEAM minus AQMC



Team 24: ATEAM minus CTEAM



EXTERNAL DISTRIBUTION:

Department of Energy  
1000 Independence Ave SW  
Washington, DC 20585  
Attn: William Reed, DP-51  
Jamileh Soudah, DP-51  
Ed Pate, DP-51  
Njema Frazier, DP-51  
Dimitri Kusnezov, DP-51

University of California  
Lawrence Livermore National Laboratory  
7000 East Ave.  
P.O. Box 808  
Livermore, CA 94550  
Attn: E. Dube, MS L-095  
Roger Logan, MS L-125  
C. Nitta, MS L-096  
N. Storch, MS L-303

Los Alamos National Laboratory  
Mail Station 5000  
P.O. Box 1663  
Los Alamos, ANM 87545  
Attn: M. Peterson  
L.Cox  
K. Koch, MS F652  
J. LaGrange, MS D445

DISTRIBUTION:

1	MS-0139 M. Vahle 9900	1	MS-0660 A. Treadway, 9519
1	MS-0139 R. Thomas 9904	1	MS-0661 G. Rivord, 9510
1	MS-0316 A. Lorber, 9141	1	MS-0739 N. Bixler, 6415
1	MS-0316 C. Ober, 9233	1	MS-0739 M. Young, 6415
1	MS-0316 T. Smith, 9233	1	MS-0759 J. Fernandez, 5845
1	MS-0316 S. Hutchinson, 9233	1	MS-0819 T. Trucano, 9211
1	MS-0429 J. Rottler, 2100	1	MS-0819 S. Petney, 9231
1	MS-0453 H. Abeyta, 2101	1	MS--0819 T. Voth, 9231
1	MS-0481 W. Moffatt, 2114	1	MS-0819 E. Boucheron, 9231
1	MS-0481 H. Radloff, 2114	1	MS-0819 K. Brown, 9231
1	MS-0482 S. Lott, 9905	1	MS-0819 S. Burns, 9231
1	MS-0482 R. Paulsen, Jr., 2109	1	MS-0819 D. Carroll, 9231
1	MS-0525 C. Bogdan, 1734	1	MS-0824 J. Moya, 9130
1	MS-0525 R. Heath, 1734	1	MS-0824 A. Ratzel, 9110
1	MS-0525 T. Russo, 1734	1	MS-0825 W. Rutledge, 9115
1	MS-0525 R. Sikorski, 1734	1	MS-0825 W. Wolfe, 9115
1	MS-0525 L. Waters, 1734	1	MS-0826 W. Hermina, 9113
1	MS-0525 S. Wix, 1734	1	MS-0826 K. Belcourt, 9143
1	MS-0626 M. Blackledge, 12326	1	MS-0826 D. Brethauer, 9143
1	MA-0638 D. Knirk, 12326	1	MS-0826 K. Copps, 9143
1	MS-0638 D. Peercy, 12326	1	MS-0826 C. Forsythe, 9143
1	MS-0660 L. Bonano, 9515	1	MS-0826 J. Rath, 9143
1	MS-0660 L. Lang, 9515	1	MS-0826 J. Stewart, 9143
5	MS-0660 K. Byle, 9519	1	MS-0826 W. H. Walker, 9143
1	MS-0660 D. Cuyler, 9519	1	MS-0827 K. Aragon, 9143
1	MS-0660 D. Eaton, 9519	1	MS-0827 M. Hamilton, 9143
20	MS-0660 M. Ellis, 9519	1	MS-0827 H. Edwards, 9143
1	MS-0660 J. Larson, 9519	1	MS-0827 T. Otahal, 9143

20	MS-0827 J. Zepper, 9143	1	MS-0847 R. May, 9126
1	MS-0828 K. Dowding, 9133	1	MS-0847 R. Kerr, 9226
1	MS-0828 M. Pilch, 9133	1	MS-0847 S. Mitchell, 9226
1	MS-0828 B. Oberkampf, 9133	1	MS-0847 R. Leland, 9226
1	MS-0834 J. Johannes, 9114	1	MS-0893 R. Brannon, 9123
1	MS-0834 M. Hopkins, 9117	1	MS-1110 M. Alleva, 9211
1	MS-0835 S. Bova, 9141	1	MS-1110 T. Barth, 9214
1	MS-0835 R. Cochran, 9141	1	MS-1110 P. Bochev, 9214
1	MS-0835 M. Glass, 9141	1	MS-1110 D. Day, 9214
1	MS-0835 G. Homicz, 9141	1	MS-1110 J. Delaurentis, 9214
1	MS-0835 R. Lober, 9141	1	MS-1110 M. Heroux, 9214
1	MS-0835 P. Sackinger, 9141	1	MS-1110 S. R. Lehoucq, 9214
1	MS-0835 J. Strickland, 9141	1	MS-1110 D. Womble, 9214
1	MS-0835 S. Subia, 9141	1	MS-1111 K. Devine, 9226
10	MS-0835 M. McGlaun, 9140	1	MS-1111 R. Heaphy, 9226
1	MS-0835 S. Kempka, 9141	1	MS-1111 M. Heroux, 9226
1	MS-0835 J. Hales, 9142	1	MS-1135 L. Gritzko, 9132
1	MS-0835 J. Peery, 9142	1	MS-1135 S. Heffelfinger, 9134
1	MS-0836 E. Hertell, 9116	1	MS-1137 G. Froehlich, 6536
1	MS-0836 R. Griffith, 9117	1	MS-1137 A. Hodges, 6536
1	MS-0836 S. Domino, 9141	1	MS-1137 H. Ogden, 6536
1	MS-0841 T. Bickel, 9100	1	MS-1137 M. Tebo, 6536
1	MS-0847 S. Attaway, 9142	5	MS-1137 M. Williamson, 6536
1	MS-0847 M. Bhardwaj, 9142	1	MS-1141 T. Vanderbeek, 6433
1	MS-0847 M. Blanford, 9142	1	MS-1146 P. Cooper, 6422
1	MS-0847 A. Gullerud, 9142	1	MS-1146 P. Griffin, 6422
1	MS-0847 M. Heinstein, 9142	1	MS-1152 M. Kiefer, 1642
1	MS-0847 S. Key, 9142	1	MS-1166 G. Scrivner, 15345
1	MS-0847 J. Koterak, 9142	1	MS-1179 L. Lorence, 15341
1	MS-0847 J. Mitchell, 9142	1	MS-9005 E. Cull, 8260
1	MS-0847 K. Pierson, 9142	1	MS-9042 M. Horstemeyer, 8728
1	MS-0847 V. Porter, 9142	1	MS-0612 Review and Approval Desk, 9612 for DOE/OSTI
1	MS-0847 T. Preston, 9142	1	MS-9042 C. Moen, 8728
1	MS-0847 G. Reese, 9142	1	MS-9217 C. Aro, 8920
1	MS-0847 G. Sjaardema, 9143	1	MS-1186, T. Mehlhorn, 1674
1	MS-0847 C. Stone, 9142	1	MS-9217 R. Tuminaro, 9214
1	MS-0847 J. Swegle, 9142	1	MS-9018 Central Technical Files, 8945-1
1	MS-0847 T. Walsh, 9142	1	MS-0899 Technical Library, 9616
1	MS-0847 H. Morgan, 9120		
1	MS-0847 K. Alvin, 9124	2	
1	MS-0847 M. Eldred, 9211		
1	MS-9007 D. Henson, 8200		