AAD Guidelines

Date: May 16, 2010

Table of Contents

1. PURPOSE / GOALS

The purpose of this document is to provide *general* guidelines for the ~~Orion~~ AADs authors/architects to ensure the documentation is consistent ~~and cohesive~~ between the various AADs. Also, this ~~seems like~~is a good starting ~~point~~ place to address some of the general comments ~~that I have~~ received from AAD reviewers. The intent is not to transform ourselves into the *same* "Technical Writing Guru" – we still want to be autonomous but establish some basic guidelines.

The benefits are three-fold: (1) it will help the reader consume the document more easily/quickly; resulting in a better feedback loop between the author and the reader (2) the consumers of the document (developers, testers, architects) will have the same understanding (3) the more precisely and accurately the requirements are written now, the less arguments and re-work will occur in the long run~~arguing we will do in the long run~~

In addition to the guidelines, the initial portion of this document tries to lay out the process involved in being an AAD author. In addition to the documentation guidelines, there are a sequence of **best practices** associated with AAD templates, ClearCase, ClearQuest, Agile, and the review process. This is the mechanism/process being followed for all Orion documents for consistency purposes. Now that all this information is documented, it can be easily refined.

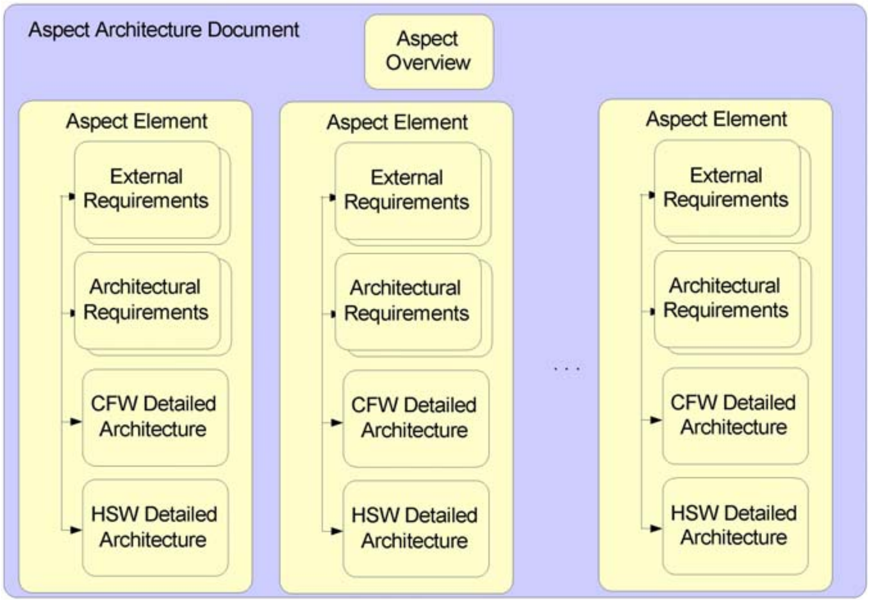
The goal of this document is not to duplicate information that is available elsewhere; in fact, it is preferable to reference those existing sources than re-hash it in this document.

2. AUTHORING AADS

2.1 AAD References

First off, below is existing content on AADs:

- AAD Cookbook
- Slide set from External CFW Dept Meeting



Aspect Architecture Document
Aspect Overview
Aspect Element
External Requirements
Architectural Requirements
CFW Detailed Architecture
HSW Detailed Architecture
Aspect Element
External Requirements
Architectural Requirements
CFW Detailed Architecture
HSW Detailed Architecture
. . .
Aspect Element
External Requirements
Architectural Requirements
CFW Detailed Architecture
HSW Detailed Architecture

Figure 1: AAD Documentation Structure

2.2 AAD Distribution List

There is a esg-aad-authors distribution list which is mentioned in the Getting Started help file at AAD Cookbook. This distribution list is used to notify authors of critical tool updates, especially for the Word editing style sheets, which requires users to explicitly run a setup script.

2.3 Authoring Tools

AAD authors fall in the following 2 broad categories:

1) If you are responsible for an entire AAD (or an aspect element of an AAD) as well as the document review cycle, then you need access and understanding of Rational ClearCase (CC), Rational ClearQuest (CQ), Agile, Adobe Professional and AAD Template.
2) If you are responsible for a portion of an aspect element and providing content to an architect, then please work with the architect to determine the best way to collaborate. There are at least a couple of options on how to do this:
    a. Provide content in normal Word/text documents and have the architect convert it to the AAD template.
    b. Provide content using the AAD template and this requires installation and understanding of Rational ClearCase and AAD template.

Depending on the type of an AAD author (see above), you will need the following tools:

1) Rational CC – send an email to rt@lsi.com to set up an account. This is used as the source/version control management system.

2) Rational CQ – send an email to rt@lsi.com to set up an account. This is used as the tracking portion of the source control management.

3) Agile – This is needed to post the documents for review

4) Adobe Professional – After getting approval from your manager, you can download this from PC Depot. Review enabled PDF files are used for review purposes.

5) AAD Template – this infrastructure exists in CC and is used to generate all architectural documents. Elaborate instructions are located at AAD Cookbook

3. AAD ORGANIZATION

There are multiple ways to organize an AAD. For example, you can have 1 AAD with 1 aspect element or 1 AAD with multiple aspect elements.

In order to ensure AAD/AspectElements are appropriately mapped, the following items should be considered:

- Scoping occurs at the Aspect Element level.
- Consider parallelism – can some content be designed/reviewed/developed independently than other content? If so, make them independent aspect elements to facilitate parallelism.
- The rule of thumb so far has been that there is 1 DA (per team, such as CFW, NAS, BVL) per aspect element, regardless of the number of ERs. [Exception to this model is Serviceability AAD – System FW Upgrade Aspect Element]
- Review and coordination occurs at the Aspect Element level, as there is usually 1 DA per aspect element. This is not true for Aspect Elements that have multiple DAs, such as Serviceability AAD – System FW Upgrade Aspect Element.
- Link in all the related (existing or new) ERs to the Aspect Element in order to maintain context for the architecture.
- AADs/Aspect Elements should be organized such that architecture can be developed iteratively to enable engineering to make maximum-possible progress on the product.

Example:

HIC Support for Pikes Peak

Option 1:

- HIC Support Aspect Element
  - FC HIC ER
  - iSCSI HIC ER
  - 1 DA

Option 2:

- FC HIC Support Aspect Element
  - FC HIC ER
  - 1 DA
- ISCSI HI Support Aspect Element
  - iSCSI HIC ER
  - 1DA

Besides the overhead associated with having 2 aspect elements, option 2 is probably the better choice since it allows independent review/coordination/development (enables parallelism) and scoping makes sense at each HIC level.

4. DOCUMENTATION GUIDELINES

~~1.1~~4.1 General Remarks

- Use the AAD template – see instructions at AAD Cookbook
- Consistent terminology across the AADs (remember, the readers are working on multiple releases at any given time so, it's important the terminology/terms are the same).
  - Use IOVM instead of DPL. Since the VxWorks VM pertains to both DPL and BVL, using DPL is technically incorrect. Of course, use the terms DPL and BVL when specifically identifying those layers. Preference is IOVM over IO VM, although, both are inter-changeable.
  - Use "Service VM" or "SVC VM".
  - Use "NAS VM" and if need to be more specific "OnStor Cougar NAS VM"
- In order to clearly articulate the content to the reader, avoid synonyms as much as possible
  - Use Domain0 (instead of Dom0 or Domain-0)
  - Use DomainU (instead of DomU or Domain-U or Guest Virtual Machines)
  - Orion vs. USP vs. OSA vs. PP
    - Use OSA when talking generically about the extensible virtualization architecture
    - Use USP when talking generically about OSA with file integration
    - Use Orion when you are specifically talking about "first release of OSA"
    - Use PP only when you are talking about "first hardware platform for OSA"
    - Use the terms OSA and USP in ERs/DAs but Orion and PP in DAs only
- Use pronouns ("it" / "this" / etc…) sparingly and carefully – it often results in ambiguity
- Only use 3rd person to specify any architectural requirements/details. Use of "We, I, Us" etc should be avoided
- ~~Provide responses to review comments in the PDF and upload to Agile; mention that in the review emails. You could also send the review comments specifically to the folks who reviewed the document.~~
- Please refrain from copying information/material from existing ~~FFDs~~ ERs and FAMs to the AAD. Instead, put in a reference to that document in Agile. This eases maintenance of documents. Else, a change in the original document also mandates a change in the document where the information is reproduced. Note that related ERs should be part of the Aspect Element – see section 3.
- As a suggestion, cross reference should refer to previous sections and not later sections. Be careful with cross references within ERs – see 4.3.

~~1.2~~4.2 Aspect Overview

Note that this is often referred to as "AADOverview"

- The Aspect Overview section should provide context of the AAD from Orion point of view, meaning how that container plays out in the big picture. Then, it should have a high-level description of the elements within that AAD - that basically tells the reader how the AAD is organized so they know what to expect.

- Capture “Open Issues” in the Overview document. Introduce sub-sections for each Aspect Element and denote issues under those sub-sections.

- Document all the mentioned documents from the ER/AR/DA in the “Related Documents” section in the Aspect Overview

- Under the “Open Issues” section, add a table to denote the AADs that will result in updates on the AAD being worked. The intent of this table is to aid all Orion authors to communicate functional ownership to ensure essential architecture is addressed. For example, in the Hypervisor Overview Section, it is denoted that “Virtualized Hardware Resources Element of the Hypervisor AAD is dependent on the following AADs/functionality, which will result in updates to that aspect element when that content is stabilized”.

  Example:

Functionality	**AAD Element ~~and area that will address the indicated functionality~~that requires it**	**Affected ~~Virtualized Hardware Resources Functionality~~Hypervisor AAD Element that needs updated**
IP Address Failover with regards to the Centralized Syslog Server	Serviceability AAD - Centralized Event Logging	~~Networking portion~~ Virtualized Hardware Resources

- Add a section/table at the same level as “Open Issues” to ~~document~~ denote functionality that is referenced in the AAD being worked that belongs in other AADs. The hope is to help readers reference other documents easily.

  Example:

Functionality	**Owned AAD Element**	**Referenced in the following Hypervisor AAD Element**
Customer Replaceable Units (DIMMs, Flash drives)	Pikes Peak Hardware Platform AAD	Virtualized Hardware Resources - Block-Only vs. Block+File operational mode

- Finally, add a section at the same level as “Open Issues” to ~~document~~ denote functionality that has been introduced in the AAD being worked that will result in additional Linux packages for that VM image.
  ~~For~~ E~~e~~xample:

Functionality	**Affected Virtual Machines**	**Referenced in the following Hypervisor AAD Element / Notes**
Heartbeat Mechanism (OpenAIS)	Domain0, Service VM, NAS VM	Virtual Machine Management

- Existing/new related ERs should be integrated into the Aspect Element for content purposes. In order to ease the review process, the overview document should clearly state the ERs or section of ERs that pertain to the DA section and the reason why the others do not.

4.2.1 AAD Template Metadata for AspectOverview.xml file

“Aspect/ElementName” and “Document Title” should state “<AADName> AAD Overview”. For example, “Pikes Peak Hardware Platform AAD Overview”.

There is no need for an Agile document number for an Overview so “Document Number” should be “00000-00”.

~~1.3~~4.3 Element Requirements

Here is an URL that talks about how to write *good* requirements ~~–~~ – http://www.ibm.com/developerworks/rational/library/5170.html - it is a very concise article. This should be useful to all the AAD authors~~, as none (other than Martin/Aaron) have written requirement documents before (at least not here at LSI). ~~.

Element requirements (ElmR) consist of external requirements (ExtR) and architectural requirements (AR). Below is an email from Keith Holt denoting the different between ExtR and AR:

In the old documentation model, an FFD specified externally visible requirements. Being “externally visible” does not imply that a requirement is directly visible to the end user, however. For example, there is a requirement in the persistent cache backup FFD to ensure that stale data from a removable flash drive does not get written on top of the existing user data. This is an externally visible requirement because failure to comply with that requirement would result in data corruption. However, this FFD also has details on how to accomplish this that probably crosses the line into implementation detail.

As another example, we may decide that in order to meet our data availability requirements, we will implement a watchdog timer. In this example, the externally visible requirement is to maintain data availability. While the use of a watchdog timer is an implementation detail, it is an architecture requirement that we need to ensure is implemented and tested.

With the Orion architecture, many of the requirements regarding virtual machine management fall under the definition of “architecture requirement.” These requirements are needed to partition functionality and to create an infrastructure for adding additional functionality by integrating applications that were not developed specifically for our environment. As such, these requirements need to be specified, implemented and tested just like externally visible requirements. Since it is not appropriate to include this level of detail in a Software Functional Specification, we have decided to add an additional document type to the AAD document hierarchy, the architecture requirement (AR) document.

An AR document will contain topic IDs with testable requirements, the same as an ER document (external requirement, sometimes referred to as element requirement). We discussed allowing architecture and external requirements to be mixed within the same document, but felt that this would create too many problems with cross-references since authors would have to be very careful not to insert a cross-reference from an ER to an AR. For now, we will create an additional top-level directory in the AAD project for ARs. An AR document will contain only architecture requirements, while an ER document will contain only externally visible requirements like this typically found in FFDs.

AR vs. ER usage model:

The full specification for an element then, may include both external requirements (ER) and architecture requirements (AR). Going back to the original example, the details on how to check for stale data in the flash drive would probably have been included in an AR written after the ER was written. Both the ER and AR would be included in an AAD “book” along with any detailed architecture documents.

Another example is discrete line management. Some aspects of this might be described in an ER, others in an AR. If a controller is held in reset, that is most definitely externally visible behavior. In this case, the ER would require a failed or non-responsive controller to be held in reset, and for that condition to be reported in the GUI. The AR would contain details on how the discrete lines are monitored and asserted to accomplish this, such as the use of the early warning indication provided on the Alternate NMI line.

Sometimes, it may be difficult to distinguish between external and architecture requirements, but the distinction is probably not critical in these cases. For example, the reporting requirements for the discrete line test would definitely be described in an ER. Additional details on the discrete line test could probably go into either an ER or and AR. We are not creating an AR just so that we can manage another document type. The AR is being created so that testable architecture requirements can be documented in living documents rather than in point-in-time detailed architecture documents.

- Use “Architecture Note” for commenting items such as, where did the requirement originate and what is the priority of the requirement?
- Avoid providing detailed architecture or design details within the ER section. They are for the DA section. In case some design details are necessary in ER, they can be used in conjunction with the “Architecture Note” tag.
- Use “Architecture Note” to indicate how the ER deviates from the PR.
- Understand the difference between a “subsection” and “named content block” – see the instructions on page 1 of the ER template
- Follow a consistent structure/organization style
- Avoid ambiguity (do not use words such as *extremely*, *numerous*)
    - Be specific about requirements
    - Separate requirements from commentary (Use “Architecture Note” for commenting items such as, where did the requirement originate and what is the priority of the requirement?)
- Use simple sentences with clear meaning
- ER should be written in present tense since when a user reads the document, the requirements should sound as though they are currently implemented.
    - Past: VM was directed to handle mgmt function.
    - Present: VM is directed to handle mgmt function.
    - Future: VM will be directed to handle mgmt function.
- In the “Administrative and Configuration Interfaces” section, think about both CLI and UI.
- Also, think about Serviceability approach in terms of “what can go wrong”, “how do we know when that happens” and “what can be done in that case”.
- Need to be careful with the type of content in the ERs – for example, an ER that talks about SATA Flash drives should not associate it with controller that supports those Flash drives. Eventually, another controller will support those Flash drives which will make that ER out-of-date. Also, ERs should not mention SATA flash drives in non-SATA flash ERs as the ER will be outdated once we move to some other interface or storage device in a future release.
- It is better to be as general as possible in an ER when talking about *dependencies*. For example, you should say “mgmt application” vs. “SANtricity” or EMW. In fact, do not denote AMW or EMW in the ER.
- Peruse the ERs/ARs to determine if an existing ER/AR should be updated instead of creating a new ER/AR. If there is an existing related ER/AR, then you can vary sections in that document or if enough items are different, then you can vary the entire ER/AR.
- Given that sections in an ER can be varied, cross section within ERs should be avoided.

4.3.1 External Requirements

- ExtR will contain externally visible behavior
- Remember that ExtR are read by the customer so information that is not meant to be seen by the customer should not exist. That information could be made part of an Architecture Note.
- Even though an AR is not seen by the user, dependencies should be carefully idenitified. For example, mgmt dependencies such as AMW/EMW should not be mentioned either.
- The virtual machine names or their specific responsibilities (such as Domain0's watchdog timer responsibility) should not be mentioned in the ExtRs since it's an appliance environment. However, high-level 'functional' responsibilities can be noted such as "block or RAID function/service", "file function/service", "mgmt function/service", etc…
- Don't forget information/critical events for Orion

4.3.2 Architecture Requirements

- It is okay to mention the virtual machine names in the ARs.
- Topics such as partitioning of the flash drive or virtual machine management should be part of an AR since there is no external behavior associated with this information, although, there are architectural requirements that are pertinent to the development and product test organizations.

4.3.3 ER & Section level variation

The ERs should have element level variation. On the first page of the ER, under "Element Information", the green INCLUDE should contain the ER top-level variation. It should contains no spaces. For example, "INCLUDE: GlenCoveHICSupport". This top-level variation needs to be introduced in the variation files in ClearCase. See the attached email from Patrick Flynn and also pertinent information at AAD Cookbook:

When trying to build an SFS I've noticed that there are several Variation Point Names that are not being associated with the Variation management files in CQ. Here are the rules for variation management:

1. Create a Plan Request Task for the variation changes and associate it with the PR you are working on
2. Create a unique variation name and associate it to the section using the variation tags (note that the top level variation point associated with a file needs to follow the same process as section based variation names)
3. Check out the VPDefinitions.xml file in the AAD\Variation Management directory Add the variation name to the file using the previous entries as an example. It's worth the effort to ensure the variation point defined in the document and the one in the VPDefinitions file are identical.
4. Check out the RequirementsMappings.xml file in the AAD\Variation Management directory
5. Add an entry associating the variation name to a PR number using the previous entries as an example. Again, it's worth the effort to ensure the variation names are identical.
6. If the PR requires a new file to be defined, check out the MasterAADList.xml file in the AAD\Variation Management directory and add the new filename and its path using the previous entries as an example. Note that this file defines the layout of the SFS document, so take care to place the file under its appropriate Feature Group Name.
7. Check the files in and deliver them as soon as possible so they will be available for others to modify.

If you have any questions, let me know.

Section level variation can also exist and for OSA, that is defined as “OpenStorageArchitectureSupport”. Whenever this is used, ensure the appropriate variation files are checked-out and the relevant PRs are associated with this variation.

4.3.4 AAD Template Metadata for ER.xml file

“Aspect/ElementName” and “Document Title” should state “<ERName>”. For example, “Glen Cove iSCSI Host Interface Card Support”.

ERs are independently uploaded to Agile outside of the AAD documentation, therefore, the “Document Number” should be specified as well.

4.4 Detailed Architecture for CFW

- Differentiate/highlight between IOVM and non-IOVM work items for the core asset sections
- Differentiate between OSA work and Orion/USP work. There is some work items which are specific to the File integration. However, there are some work items for all OSA platforms. Authors should be able to differentiate that. This will include appropriate variation points.
- When defining a new core asset, please refer to the nomenclature of existing core assets.
- When defining new MELs, please define which core asset should own that MEL event.
- Denote research/prototype material in an Appendix section and leave the DA for the crisp details of the design.
- Ensure components from other AADs are utilized, instead of introducing new components.
- Consider the environment of the functionality:
    - Which VM(s) does it run in?
    - Main memory / flash disk memory / IOVM disk requirements, if significant
    - Any assumptions about how internal/external network is configured
    - How does it persist configuration
    - Any interfaces to other VMs and if so, what transport/mechanism
- To avoid ambiguity of referring to components that have the same name in multiple VMs, [VM_NAME]<Component_Name> notation should be used. For example:

    [IOVM] RMI will call [Domain0] RMI to transfer data.

4.4.1 AAD Template Metadata for DA.xml file

“Aspect/ElementName” and “Document Title” should state “<DAName>”. For example, “Glen Cove iSCSI Host Interface Card Support”.

There is no need for an Agile document number for an DA so “Document Number” should be “00000-00”.

5. AGILE GUIDELINES

For the AAD process, there is a need for Agile document numbers for the following reasons:

1) External Requirement and Architecture Requirement document types in order to allow independent viewing/reading of requirements. This is something we wanted to maintain in case folks wanted to view requirements outside of any program/AAD/etc…
2) Aspect Element for review purposes

5.1 Posting Documents to Agile

The "best practice" guidelines for reviewing AAD files should be to post in Agile, then send a link that takes you directly to the change bar version of the PDF file to be reviewed. Current practice is to include a link to the document object itself, when requires reviewers to first select the latest version of the document object, then select the Attachments tab, then select the attachment to be opened.

Since a clean version and a version with comments and replies are also attached to the document object, a link to the document object can be included below the attachment link, similar to what is shown below.

In addition, the best practice for attachment management should be to use the Agile attachment check out/ check in facility when posting a new version of an attachment (e.g. version B.3 replaces B.2). This would nominally show only 3 attachments for each document object (clean, change bar, change bar (n-1) with comments and replies) rather than a dozen or more attachments for all revisions. Older versions can still be obtained using the Show Versions button. Follow the last link to the T10 PI FFD for an example.

Example using T10 PI FFD:

Direct link to change bar version PDF file in Agile:

https://agile.lsi.com/Agile/link/SW%20Eng/34472-00/Rev/DC08225/files/FFD_T10DIFDataProtection_D5_D4_Diff.pdf

Standard link to document object in Agile (must select version in drop-down list after following link):

https://agile.lsi.com/Agile/PCMServlet?fromPCClient=true&module=ItemHandler&requestUrl=module%3DItemHandler%26opcode%3DdisplayObject%26classid%3D9000%26objid%3D158007172%26tabid%3D12%26

5.2 ERs for Agile

5.2.1 PDF Generation of ExtRs/ARs for Agile

A PDF file of the ER needs to be generated for Agile since it is a living document. It does not have to be a reviewable (aka, comments enabled) PDF file.

After following the guidelines in section 4.3.4, run "buildAAD <ER>.xml <ERName>.pdf". The generated PDF files should be uploaded to Agile.

5.3 AspectElement for Agile

- Name the Agile doc/DCO as “AADName” AAD – “Aspect Element Name” (do not put “aspect element” at the end of it)

5.3.1 PDF Generation of Aspect Element for review purposes

A PDF file of the AspectElement needs to be generated for review purspoes. It needs to be a reviewable (aka, comments enabled) PDF file.

After following the guidelines in sections 4.2.1, 4.3.4, and 4.4.1, create a control file called “<AspectElement>FileList.xml” (such as “GlenCoveHICSupportFileList.xml” to be checked into ClearCase under the appropriate “ArchitectureAspects” directory.

With “Pikes Peak Hardware Platform” AAD and “Glen Cover HIC Support” Aspect Element as examples, the contents of the control file for review purposes should be as follows:-

```
<AADFileList>
    <DocumentInformation>
        <CopyrightYear>2009-2010</CopyrightYear>
        <DocumentTitle>Pikes Peak Hardware Platform AAD</DocumentTitle>
        <DocumentSubtitle>Glen Cove HIC Support</DocumentSubtitle>
        <DocumentHeader>Pikes Peak Hardware Platform AAD - Glen Cove HIC
Support</DocumentHeader>
        <DocumentNumber>47565-00</DocumentNumber>
    </DocumentInformation>

<FrontMatter>
    <Ref srcfile="..\..\DocumentTools\AAD_FrontMatter.xml"/>
</FrontMatter>

<AspectOverview heading="Pikes Peak Hardware Platform AAD - Aspect Overview">
    <Ref srcfile="AAD_PikesPeakHardwarePlatform.xml" />
</AspectOverview>

<AspectElement heading="Pikes Peak Hardware Platform AAD - Glen Cove HIC
Support">
    <Ref srcfile="ER_HIC_Glen_Cove_2x10GbiSCSI.xml" heading="Glen Cove iSCSI
HIC Support - External Requirements"/>
    <Ref srcfile="DA_DPL_HICSupport.xml" heading="Glen Cove HIC Support -
Detailed Architecture"/>
```

```
</AspectElement>

</AADFileList>
```

After the above file is created, run “buildAADBook “<AspectElement>FileList.xml “<AspectElement>.pdf”. After the PDF file is enabled for commenting/reviewing (see section 8), it should be uploaded to Agile.

5.3.2 Review Responses

- Provide responses to review comments in the PDF file and upload to Agile; mention that in the review emails. You could also send the review comments specifically to the folks who reviewed the document.

6. CLEARCASE GUIDELINES

Clear Case Remote Client (CCRC) installation instructions are found at the AAD Cookbook site.

All ExtRs are checked into the AAD/ElementRequirements/<TopicArea>/ and ARs into AAD/ArchitectureRequirements/<TopicArea>/ with no further sub-directory.

AADOverview and DAs are checked into AAD/ArchitectureAspects/<AADName>/, with no further sub-directory except for the Graphic files. This is the existing guideline but it can be modified if there is agreement – the key point is that whatever process we use should be consistent.

Please do not prevent ClearCase use for others – if you need to add new files/directories then create a “PlanReqTask”, create new files/dirs, then check-in/deliver. Then, you can create a “PlanReqTask” to check-out the files you need to work on. This applies to the 3 variation files in the AAD/VariationManagement directory as well as those files are frequently sought after.

7. CLEARQUEST GUIDELINES

Here are CQ/AAD Training slides: AAD Review Process

Below table captures AAD CQ records from an Architect's viewpoint:

Record Type	**What it contains**	**What should an architect do?**
AAD	It contains record associations for AADOverview and AADElement(s).	Create the necessary AADOverview and AADElement .associations
AADOverview	Association with an AAD. It also has an option to add/create PlanReqTask.	Add "Description" field under "Main". Generate a PDF file with the Overview and attached in the "Attachments" tab. See 7.1.1 for detailed information. Create a PlanReqTask to check in the Overview.* Transition to ProdMgmtReview.
AADElement	Associated AAD PR associations Scoping Request associations	Attach PRs associated with the AspectElement. Also, attach ScopingRequests If no further revisions left of any of the associated documents, transition to Complete.
ElementRequirement	Associated AAD Element It also has an option to add/create PlanReqTask.	Enter Agile Document # and Revision for the ER – see section 5.1 . Create a PlanReqTask to check in the ElementRequirement.* After CC check-in/deliver, transition to Review or Conclude.
DetailedArchitecture	Associated AAD Element It also has an option to add/create PlanReqTask and DevelopmentRequests.	Enter Agile Document # and Revision - see section 5.2 Create a PlanReqTask to check in the ElementRequirement.* After CC check-in/deliver, transition to Review or Coordination.

* = Note that if you have multiple document types to check into ClearCase, you can use 1 PlanReqTask for all of them and associate them with each of the respective CQ records.

Below is some direction from Bill Lomelino

Now, what records get updated as you develop the docs and then send them out for review depends on what you working on. So think of it as bookkeeping PER section of an AspectElement released for review. Each section will have a corresponding record in CQ and you will need to transition each record that is being updated. So in your case if you are updating the ER and DA then you would make the appropriate transitions of the corresponding ER and DA record in CQ. So the more ERs you update in a given release of the Aspect Element means more ER records in CQ to update accordingly. Now if you release an AspectElement for review and you only update the DA then you only deal with the DA record in CQ.

7.1.1 PDF Generation for CQ AADOverview Attachment

A PDF file of the AspectOverview needs to be generated for Product Management's approval. It does not have to be a reviewable (aka, comments enabled) PDF file.

After following the guidelines in section 4.2.1, create a control file called "AADOverviewFileList.xml" to be checked into ClearCase under the appropriate "ArchitectureAspects" directory.

With "Pikes Peak Hardware Platform" AAD as an example, the contents of this control file should be as follows:-

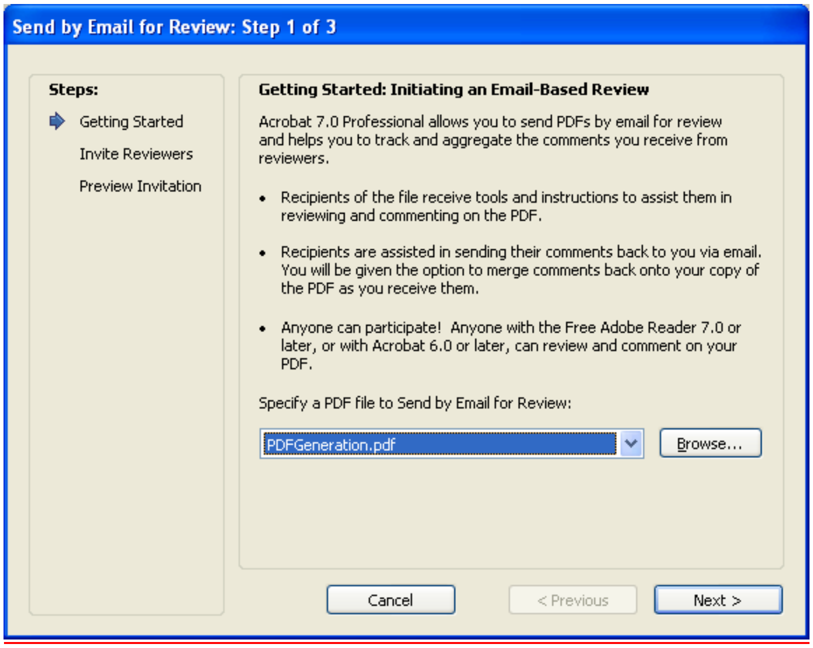
```
<AADFileList>
    <DocumentInformation>
        <CopyrightYear>2009-2010</CopyrightYear>
        <DocumentTitle>Pikes Peak Hardware Platform AAD</DocumentTitle>
        <DocumentSubtitle>AAD Overview</DocumentSubtitle>
        <DocumentHeader>Pikes Peak Hardware Platform AAD Overview</DocumentHeader>
    </DocumentInformation>

    <FrontMatter>
          <Ref srcfile="..\..\DocumentTools\AAD_FrontMatter.xml"/>
    </FrontMatter>

    <AspectOverview heading="Pikes Peak Hardware Platform AAD - Aspect Overview">
        <Ref srcfile="AAD_PikesPeakHardwarePlatform.xml" />
    </AspectOverview>

</AADFileList>
```

After the above file is created, run "buildAADBook AADOverviewFileList.xml AADOverview.pdf". The generated PDF file should be attached to the CQ AADOverview record.

8. ABODE ACROBAT PROFESSIONAL GUIDELINES FOR DOCUMENT REVIEW

8.1 Sending the PDF file for review

- After creating the PDF file from “buildAADBook”, open the PDF file that is meant to be sent for review.
    - Select “Attach for Email Review” from the “Comments” menubar
    - You will get the following dialog box:



Send by Email for Review: Step 1 of 3

Steps:

Getting Started

Invite Reviewers

Preview Invitation

Getting Started: Initiating an Email-Based Review

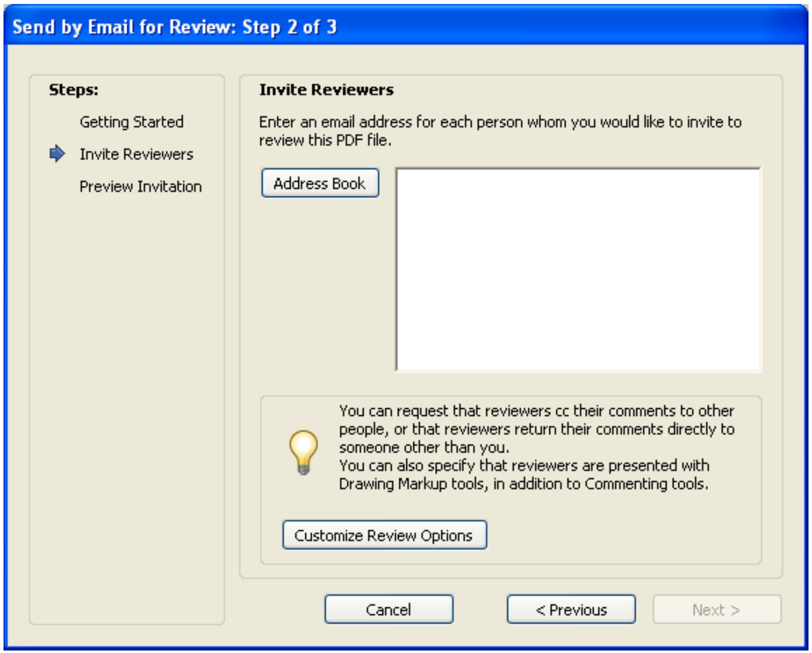
Acrobat 7.0 Professional allows you to send PDFs by email for review and helps you to track and aggregate the comments you receive from reviewers.

- Recipients of the file receive tools and instructions to assist them in reviewing and commenting on the PDF.
- Recipients are assisted in sending their comments back to you via email. You will be given the option to merge comments back onto your copy of the PDF as you receive them.
- Anyone can participate! Anyone with the Free Adobe Reader 7.0 or later, or with Acrobat 6.0 or later, can review and comment on your PDF.

Specify a PDF file to Send by Email for Review:

PDFGeneration.pdf

- You don’t need to change anything here so click “Next >” and you get the next dialog box:



Send by Email for Review: Step 2 of 3

Steps:

Getting Started

Invite Reviewers

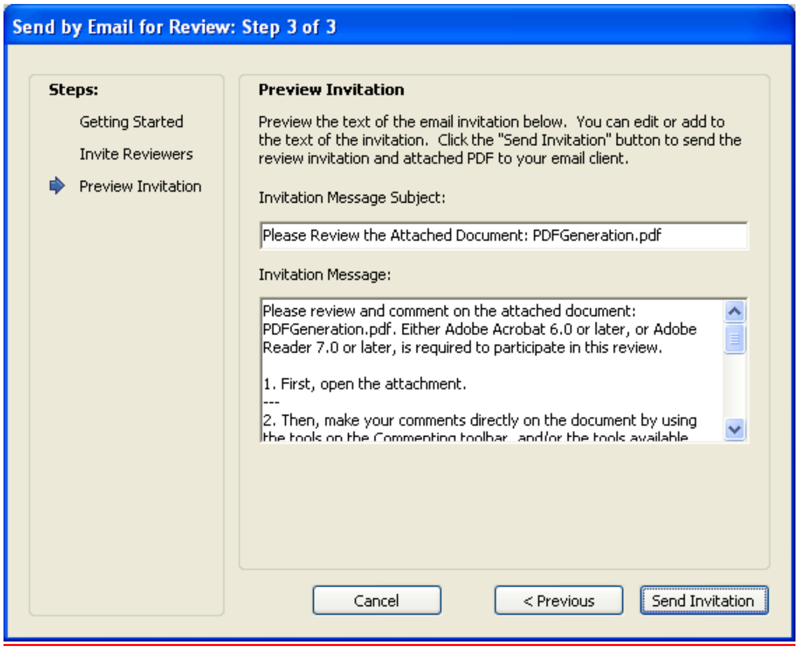
Preview Invitation

Invite Reviewers

Enter an email address for each person whom you would like to invite to review this PDF file.

You can request that reviewers cc their comments to other people, or that reviewers return their comments directly to someone other than you.
You can also specify that reviewers are presented with Drawing Markup tools, in addition to Commenting tools.

- Now, place your cursor inside the box and type in your email address (firstname.lastname@lsi.com) and then click the "Next >" button which will become visible once you have entered your email address. The next dialog box appears.



Send by Email for Review: Step 3 of 3

Steps:

Getting Started

Invite Reviewers

Preview Invitation

Preview Invitation

Preview the text of the email invitation below. You can edit or add to the text of the invitation. Click the "Send Invitation" button to send the review invitation and attached PDF to your email client.

Invitation Message Subject:

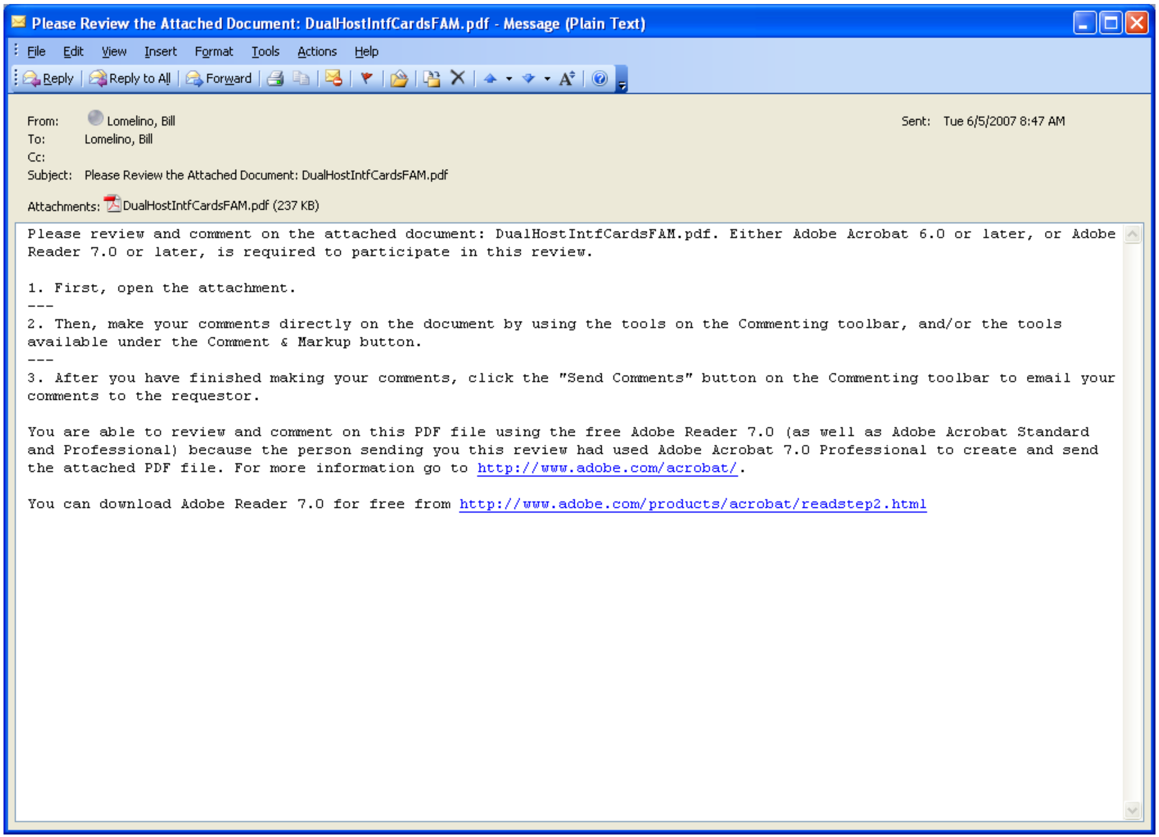
Please Review the Attached Document: PDFGeneration.pdf

Invitation Message:

Please review and comment on the attached document: PDFGeneration.pdf. Either Adobe Acrobat 6.0 or later, or Adobe Reader 7.0 or later, is required to participate in this review.

1. First, open the attachment.
---
2. Then, make your comments directly on the document by using the tools on the Commenting toolbar, and/or the tools available

- This dialog box is fine as is so hit "Send Invitation".   The send review for review portion is now finished.
- You will receive an email that looks like this:


Please Review the Attached Document: DualHostIntfCardsFAM.pdf - Message (Plain Text)
File
Edit
View
Insert
Format
Tools
Actions
Help
Reply
Reply to All
Forward

From: Lomelino, Bill
Sent: Tue 6/5/2007 8:47 AM
To: Lomelino, Bill
Cc:
Subject: Please Review the Attached Document: DualHostIntfCardsFAM.pdf

Attachments: DualHostIntfCardsFAM.pdf (237 KB)

```
Please review and comment on the attached document: DualHostIntfCardsFAM.pdf. Either Adobe Acrobat 6.0 or later, or Adobe
Reader 7.0 or later, is required to participate in this review.

1. First, open the attachment.
---
2. Then, make your comments directly on the document by using the tools on the Commenting toolbar, and/or the tools
available under the Comment & Markup button.
---
3. After you have finished making your comments, click the "Send Comments" button on the Commenting toolbar to email your
comments to the requestor.

You are able to review and comment on this PDF file using the free Adobe Reader 7.0 (as well as Adobe Acrobat Standard
and Professional) because the person sending you this review had used Adobe Acrobat 7.0 Professional to create and send
the attached PDF file. For more information go to http://www.adobe.com/acrobat/.

You can download Adobe Reader 7.0 for free from http://www.adobe.com/products/acrobat/readstep2.html
```

- So, you need to save the PDF document somewhere other than where you specified the PDF with comments enabled to be saved, one option is some folder on your desktop. The version of the PDF you saved at the beginning of this process is the "review master" and doesn't have comments enabled. SO you have to get the version of the PDF in this email out in Agile.

  After saving the PDF from the email locally, you are ready to save newly saved PDF in Agile.

8.2 Review Responses

Once reviewers send their comments, responses to their comments should be posted on Agile. Before doing so, the following guidelines should be followed for consistency:

- There is a "common legend" used for the review responses for the AADs – the following is being used for AADs:

  Legend for the responses:

  Accepted = comment results in changes/modifications to the document and will be addressed in the next revision

  Rejected = comment does not result in changes/modifications to the document

Checked = comment has been addressed in the revised version (this is used internally by the author to keep track of responses)

I embed the above (Accepted/Rejected) text into a comment at the top of the document so that reviewers (of the responses) understand this.

The ‘Checked’ item is *optional* – I use that to keep track of everything but it’s your preference if you want to do the same.