



























































## Known Architecture Risks, Assumptions, Issues and Dependencies

RAID which have a potential impact on the project must be tracked by the Project Manager in CMT. Architecture RAID items can be managed in SharePoint. Any architectural RAID which begins to impact the project must be escalated to the PM and managed on CMT. Templates for the SharePoint Logs can be found on the ‘High Level Design Templates’ page in Compass which is [here](http://hbos.intranet.local/gito/adf/COMPASS2/EAD%20Industrialised%20Design%20Pages/High%20Level%20Design%20Templates.aspx) and instructions on their use [here](http://hbos.intranet.local/gito/adf/COMPASS2/EAD%20Industrialised%20Design%20Pages/SharePoint%20Lists.aspx).

RAID Items that are architecturally significant are about e.g. if there is new or amended integration with external systems or internal critical systems or if the solution is dependent on a new piece of architecture that is still in design and so part of the Detailed Architecture work cannot be completed.

**It is expected that all Architecture RAID items relevant to the Architecture baseline being submitted for Quality Gate 2 approval will have been mitigated, resolved or otherwise they must be escalated to project RAID, logged in CMT and addressed in the Project Management plan.**

Quality Gate 2 – export, from the project SharePoint list, the baseline of the Architecture RAID items that aligns with your Detailed Architecture and copy and paste below.

### Risks

|  |  |  |  |
| --- | --- | --- | --- |
| Risk Type | Definition | Impact on Accounting Hub | Probability of Occurrence and mitigation |
| Maintenance Risk | The amount of change that needs to be done in order to introduce a new functionality | The application (Accounting Hub, AMD, and the File Processing) is split into the following three classes of components:   1. Kernel 2. Optional 3. Variable   Accounting hub / AMD and the File services is built on the Microkernel architecture pattern.  The frameworks used are well known in the industry and have support and maintenance contracts | Medium 🡪 Low  ( the established design governance and development checks must be followed before a maintenance fix is introduced) |
| Reliability Risk | How reliable is the application in terms of availability, recoverability or responding to technical and business faults | Accounting Hub (Message Broker component, AMD, and File Processing) is multi-instance. The code does not have any server affinity. Hence all messages can be received in all nodes and responded from all nodes. Moreover the request and the corresponding response can take different paths. There is no need to replicate session state hence we can add / remove nodes when we want without impacting the overall service. The added nodes can start to function instantaneously. If we want to increase the availability further my suggestion is to make the DR active.  The file processing component will have the framework manage the state of a job. If the job fails in a node then the framework will restart the job.  As part of this project Accounting hub will introduce a request tracking component to track messages which has been gobbled by an instance and the instance has failed before it has processed it and sent it to any product platforms. | Low  The MTBF is considered to be low doe to node agnostic nature of the design. |

### Assumptions

The assumptions are:

1. The existing interfaces on the hHBOS product Platforms will be reused without any changes.
2. Accounting hub will not have any implicit postings it will generate the posting entries based on the account model configured within AMD:GetNPA
3. The account model will be provided by the Payment engine projects. The model would have the primary as well as the compensating entries. If there are no compensating entries Accounting hub will not be able to complete the rollback of the posting transaction.
4. The Account model assumed is based on the FPS model for hHBOS. The principal entry is posted to the hHBOS Product platforms (wCBS, TD01, IF-TD01, IF-Lynx, NCA) while the settlement is passed to rCBS. The settlement is manually carried out between rCBS and the hHBOS product platforms.
5. There is no checking of CHAPS payments for insufficient funds. It is a business decision as to whether a payment should be made. If CHAPS payments are initiated from on-line banking, the on-line banking system checks the account has sufficient funds before the payment is made. The on-line banking system also posts the transaction to wCBS rather than this being done by the CHAPS processing.
6. For this implementation the assumption is not to do stand-in for the hHBOS product platforms hence the consuming platform for Account Posting service will not set the stand-in flag. If the flag is set then Accounting Hub will ignore the value of the flag.

### Issues

1. The file and message formats are being discussed with the ADM teams. These formats will have to be verified by the Payment engine projects
2. The posting file generation schedule will be provided by the Payment engine projects.

### Dependencies

1. Payment engine projects will have to provide the NPA / Account model for each payment scenario they want to execute.
2. Payment engine changes to use Accounting hub will be outside the scope of this project. The Payment engines will have to be changed and the messages to accounting hub will be in Accounting hub’s format
3. Payment engine changes are required to consume the Accounting Hub Services. These changes are outside the project scope
4. System Integration testing will be done by the Payment engine projects as they will provide us the accounting model for the payment scenarios
5. Non-Functional Testing will be done by the Payment Engine projects as they would provide us more accurate volumes