

***V1 Data Migration Service Requirements***

**Problem**: Many clients need to combine and/or merge two or more instances of VersionOne together. This often occurs when two V1 clients merge, or different teams with a client organization merge.

**Solution**: Create a Windows application/tool that can be used to pull data from one V1 instance into another, effectively merging the data contained in both.

**Features:**

1. Migration Analyzer:
   1. Compares source/target systems for compatibility and differences
      1. V1 versions
      2. V1 API versions
      3. Custom fields
      4. List type values
      5. Configurations
2. UI control:
   1. User can select source/destination instance (and validate)
   2. User can set V1 account credentials
   3. User can select which projects(s) to migrate
   4. User can select which assets to migrate
   5. User can select which fields of each asset to migrate
   6. User can select to import open|closed|deleted assets.
3. Display UI of merge progress
   1. Use dual progress bars: one for major project/asset type, one for the number of assets
4. Generates log of merge results, list items that could not be merged
   1. Maybe as an HTML report?
   2. Kick out as Excel import sheet to trying importing later?
5. Uses Core API for all merge activities (read from one system, create in another)
6. All asset values will be preserved including state.
7. Use config database to save state (config files are evil!):
   1. Source/Target systems
   2. Option selections
   3. V1 credentials (excluding password)
8. Supports authentication through username/password, Windows auth, OAuth (future)
9. Support easily adding new V1 assets as they become available.

**Caveats:**

* Will not migrate historical data
* Need to be cautious of performance and timeouts (especially when migrating hosted instances)

**Issues/Questions:**

* Support merging custom fields? May not be possible with API.
* Need to support multiple versions of V1?
* Need to use TDD?
* Should be Windows application or more like a wizard?
* Have a “configuration analysis” report? Would show differences in list types, data types, custom fields, and other configurations between two systems. Would all show a count of all the assets that would be migrated.
* Have ability to resolve some differences? For example, add values to a list type.
* Handling duplicates…what if there are duplicate project names?
  + Could run a “precheck” to ensure that there are no conflicts with:
    - Usernames
    - Schedules
    - Projects
* Automatically perform database backup of “destination” source before attempting the migration?
* Only V1-to-V1? Or should it support migrating from other systems (Rally, JIRA, Scrumworks, etc.)
* Add ability to support transformations? (EX: This list value should become this other value in target system).

**Data to Migrate:**

The migration tool will support migration of the follow assets, in this order:

1. Users (MEMBER)
2. Teams (TEAM)
3. Member Groups (MEMBERLABEL)
4. Schedules (SCHEDULE)
5. Projects (SCOPE)
6. Programs (SCOPELABEL)
7. Sprints (TIMEBOX)
8. Feature Groups (THEME)
9. Goals (GOAL)
10. Requests (REQUEST)
11. Issues (ISSUE)
12. Templates (PrimaryWorkItem with AssetState = 200)
13. Epics (EPIC)
14. Stories (STORY)
15. Defects (DEFECT)
16. Tasks (TASK)
17. Tests (TEST)
18. Build Projects (BUILDPROJECT)
19. Build Run (BUILDRUN)
20. Change Sets (CHANGESET)
21. TestSets (TESTSET)
22. TestSuites (TESTSUITE)
23. TestRuns (TESTRUN)
24. Links (LINK)
25. Attachments (ATTACHMENT)
26. Conversations (EXPRESSION)
27. RegressionPlans (REGRESSIONPLAN)
28. Regression Suites (REGRESSIONSUITE)
29. RegressionTests (REGRESSIONTEST)
30. TeamRooms (TEAMROOM)
31. Retrospectives (RETROSPECTIVE)
32. Relationships (DEPENDENCIES|DEPENDANTS)
33. Actuals (ACTUAL)
34. Subscriptions (SUBSCRIPTION)
35. Capacity (CAPACITY)
36. Environments (ENVIRONMENT)

**Wizard Steps:**

1. WELCOME: Welcome screen
2. SOURCE: Select source instance and account
3. DESTINATION: Select destination instance and account
4. PROJECTS: Select source projects to migrate (with select all option)
5. ASSETS: Select the assets to migrate, option to include closed assets
6. ANALYSIS: Perform migration analysis and display results
7. SUMMARY: Summarize selections before migration
8. PROGRESS: Shows progress of migration in real time
9. FINISH: Final screen that displays migration statistics

**Asset Import Notes**

* To import FeatureGroups, scopes must be all active.

**Members**

* Will not import Member:20 (admin).
* Check for duplicates using Username.
* Deactivate if AssetState = “Closed”.

**MemberGroups**

* Can only be deleted in V1, will not import anything other than “Active”.
* Check for duplicates using Name.

**Teams**

* Check for duplicates using Name.
* Deactivate if AssetState=”Closed”.

**Schedules**

* Check for duplicates using Name.
* Deactivate if AssetState=”Closed”.

**Projects**

* Will not import Scope:0 (System).
* No check for duplicates.
* Deactivate if AssetState = “Closed”.

**Programs**

* Can only be deleted in V1, will not import anything other than “Active”.
* Check for duplicates using Name.

**Iterations**

* ISSUE: Will not import when schedule is NULL.
* No check for duplicates.
* Iterations can be Active|Closed|Future.

**Goals**

* ISSUE: Will not import when scope is NULL.No check for duplicates.
* Deactivate if AssetState = “Closed”.

**FeatureGroups**

* ISSUE: Will not import when scope is NULL.No check for duplicates.
* Deactivate if AssetState = “Closed”.
* ISSUE: How to preserver order (rank) on import.

**Requests**

* ISSUE: Will not import when scope is NULL.
* No check for duplicates.
* Deactivate if AssetState = “Closed”.
* ISSUE: How to preserve order (rank) on import.

**Issues**

**Epics**

* ISSUE: Cannot import Risk when epic comes from a story, does not mean the same thing.

**Stories**

* ISSUE: Timeboxes must be active to associate with a story.
* ISSUE: Epics must be active to associate with a story.