**Diplomacy**

Each player has a set of 4 index slots that contain information about the following: other specific players, other cultures, other religions, specific org structures.

In the index file, each record (8 bytes) contains the key for the ID of the appropriate player, culture, religion, or org as well as an slot value for information about that specific item.

Culture slot -

The culture slot lists the current status with the culture as a whole, then has specific exceptions that may apply.

**Events**

Events are stored in an event data file that contains relevant information for each item. Each item also has a scheduled completion time and an index link to the prior and next events to occur. If changes are made to the completion time, the links can be adjusted to point to the correct pred/successor.

Processing - Events are processed in the movement script. Each minute, the events set to occur in that minute are loaded into the script and processed. Part of the processing is a confirmation that the event finish time has not changed.

**City Interface**

* View players/characters in the city
* View/Manage resource production for the city
  + List all resource production and current rates
  + List all resource buffs and nerfs
* View/Manage city upgrades

**Tech Ideas**

Culture specific buildings

* Blacksmith, stables, etc

Culture specific characters

* Smith, Trainer, etc

Weapon/Armor upgrades (implemented with appropriate level buildings)

Culture/Happiness boosts

Religion Happiness boosts

**Foraging/Land Effects tracking**

Any time and action or event happens that affects the surrounding land, an affect shape will be created or referenced. This shape will be a list of points in the char format that will have the affect on the surrounding area as the value. Each shape will be a specified box (120 x120?) and can be merged and stacked to give a cumulative affect.

**Unit and Action "Cards"**

Each unit or action is a "card" or one time use object that can be played or deployed by the player. These can be recharged or purchased outright.

**Task/Work Processing**

Each unit that has a current, in progress task will be listed by ID In the work porgress file (workProg.dat). The unit IDs from this file will be read into the engine and the parameters for each task will be loaded. For units who are gathering resources, the parameters of the research point will also be loaded.

Gather points – read into a map where unit ID is map key and the relevent parameters are the values in an array type

Type 1 task - Adding points to a project/plot etc.

Rb = base rate of points added per minute

Bx = experience factor bonus

Et = Experience Total

El = Experience Level

El = 2\*Sqrt(Et/(24\*60))

Bx = El\*300/60 = El\*5 = 2\*5\*sqrt(Et/1440) = 10/(2\*6) \* sqrt(Et/10) = 5/6 \* sqrt(Et/10)

Points per unit of time = Rb + Bx

One unit of time = 1 Experience Point

**Worker information array**

Job Type, Experience address, Total Experience, Task End Time, Store address to add to, Resource/Target ID

**Resource information array (Type 1)**

Total points when full, recharge rate, current points, Store address to deduct from, base rate

**Project information array (Type 2)**

Total points to complete, current points, point store address to add to

**Files**

openJobs.job – Listing by unit ID of what current job the unit is assigned to

jobChange.log – Listing of changes to job orders. The engine reads up to the end and then checks for new additions to the file and checks those. The listings are the unit ID to check for a job change. That ID will be checked for a potentially new job in openJobs.job

tasks.tdt – The task detail for each unit and location