Sprint 4 -- 10 day duration (Ends 21st of April, 2017) 60 hours of work.

Basic Design

The opening parameters for the design remain the same as for the previous three sprints. One of the main tasks of Sprint 4 will be to explore the consequences of these choices more thoroughly.

Risk List

1. I don't have a solid enough understanding of how the game functions, which will make it very hard to make intelligent design decisions.
   a) This problem will largely resolve itself through my own personal tests. I'll start by creating a network that can sustain a city, either under this mechanical paradigm or some other, and then make targeted changes to that assembled network to see how those changes affect the network. The important thing isn't to take actions willy-nilly, but to think my experiments out ahead of time and have a clear purpose guiding every decision. It'll be a matter of discipline for the most part.
2. It's frustrating and disrupting to have societies accidentally ascend, especially when they form a critical part of your resource distribution network.
   a) I'll add onto the program the ability to suppress ascension for a given society, thereby mitigating the disruption that accidental ascension causes. I might also consider making changes to the needs and ascension costs of the various complexity levels so that accidental ascensions are more difficult. For instance, making it so that ascension requires a certain multiple of needs in addition to a certain multiple of production. That way, a village that's having its yellow extracted will never accidentally ascend because it's been given too much food.
3. It might be too difficult to form an infrastructure capable of sustaining a single city, let alone one that sustains multiple cities.
   a) I'll have to see how a "standard city network" looks in the digital prototype, and then experiment with variations on that structure, especially those that result in factory patterns. Whether the patterns are too complex will come to light as testing progresses, at which point we should have a solid idea of why the problems are occurring and can address them more specifically.
4. Controlling permissions and priorities on highways is not a sufficiently expressive set of verbs. Being unable to control the amount of a resource that a society can receive can cause oversupply issues that cannot be addressed.
   a) I can resolve this by giving players control over ResourceDepot capacities and Society stockpiling. For societies in particular, I can give players a set of capacity templates (accept only needs; accept needs and wants; accept needs, wants, and costs of ascension, etc) and then allow them to set capacities to be multiples of these templates. For instance, a player could decide to let a farm stockpile three production cycles worth of wants, or a village to stockpile two cycles worth of needs and one worth of wants, or to have a town stockpile up to its costs of ascension. That ought to give expressive control over the fate of a society.
5. Players might not be able to convert sections of the map into meaningful factories by which they might supply other regions.
   a) I know that Paper Prototype 2.2 had factories of some description, and the ability to export blue (which can be used to increase other production) is almost guaranteed to create factory

patterns, but I'll need to confirm that this still works as intended in Digital Prototype 2.2. The importance of this risk will become evident as other experiments continue.

6. A complex network of ResourceDepots to transfer resources long distances might take up so much of the map that it infringes on resource production and makes long transportation networks cumbersome and unappealing.
   a) While it's hard to say for sure, I feel like ResourceDepots offer an interesting tradeoff. Since land is limited, you need to decide whether to invest in additional production, or additional logistics to distribute the production. Most of my tests have, up to this point, been too limited to say for sure whether ResourceDepots take up too much space.
   b) Nevertheless, there are several ways in which we can adjust to make ResourceDepot indirection less of a problem. The easiest way is to increase the bushiness of our object graph. If there are more edges connecting the nodes, then a single ResourceDepot can serve a larger number of Societies. I might also consider allowing ResourceDepots and Societies to share spaces under certain circumstances. Perhaps farms and/or villages can share a spot with ResourceDepots, while towns and cities cannot?
   c) Another more drastic turn is to pull ResourceDepots off of the MapGraph altogether, instead letting them float freely. That would require a complete redesign of the transportation system, and should only be used as a last resort if I cannot resolve the problem in the current design space.
7. The current logic of ascension is unintuitive, especially the inability of farms to ascend and the asymmetries in the village/town and town/city ascensions.
   a) One way of addressing this is to find a single pattern of ascension that is applied to all Societies. For instance, I could make the ascension cost to a given society require all of the resources that the new society will need, including the productions of the previous level of complexity. So farms would need food to ascend, villages would need yellow, and towns white. That paradigm doesn't resolve any other problems, like the problem of ascending into a society that you can't possibly support, but it does help standardize things and make it harder to accidentally ascend a society. I don't think that paradigm would work for farms, though. That would cause farms that aren't being extracted from to accidentally ascend into a village, then descend back into a farm, which would exacerbate the supply shock problem. Better, I think, to maintain farms as a special case that needs to be generated from above.
8. The supply networks might be too simple to lead to engaging gameplay, or too complex for a beginner to wrap their head around it.
   a) To me, this question revolves around whether or not I need multiple types of society at the same level. For instance, should there be types of raw material producers other than farms (like lumber mills, mines, etc)? Are there different types of villages? That's a much larger design question. I definitely want things like that in some future version of the game, and the experiments of Sprint 4 might lead me to a position from which I can begin exploring that, but I doubt I'll address increasing complexity this sprint.
   b) On the other hand, if the system is too complex then I'll need to do as much as I can to streamline the process, including by improving the UI and changing the networks required to be more forgiving. One option might be to change the ratios of production, so that (for instance) a single farm can support slightly more than a single village, a single village slightly more than a single town, and so on. Increasing the margins for error will certainly make assembling these networks easier to accomplish.
9. The current mechanical structure of the game might constrain map design and severely limit the variety the game can offer.
   a) 8b contains some suggestions as to how one might address overconstraint. Making a city-supporting network easier to produce, and produce more surplus, ought to help loosen up the

constraints of the map and allow for more design space. If the limitations of BlobDistributor or ResourceDepot end up being a bottleneck, I can always change the logic of BlobDistributor and add DepotUpgrades to allow more resources to flow. But if I have problems with, say, ResourceDepot saturation, then I don't think any magic I can work will free up the constraints of the map. I'll need to reconsider the design from that point.

10. The conditions of ascent and descent might be frustrating to manage, especially when societies change complexities off-screen.
    a) There are a few ways I can think of to address this. One is to increase the descent timer, to give players more time to spot and react to sudden resource shortages. Another is to give the option to suppress ascension, to prevent accidental ascensions that might cause supply shocks. Another is to make it very clear when a society is stable and when it's in the process of descent. If I added some UI element, visible on the normal map, that indicated either that the society is stable or the time until it descends, then players will be much more likely to see problems in their network as they arise. Adding faculties for pausing, slowing down, or speeding up the game might also help people address problems in moments of sudden, widespread issues. And none of these changes would be particularly difficult, either.
11. It might be very difficult to determine where blobs are coming from and where they are going, especially in a complex network. Likewise, it might be difficult to determine the production and flow of resource types.
    a) One relatively simple solution is to make a given blob selectable and then draw trails as to where that blob has been. That way, players can see the path a given blob has taken. That won't be too difficult to program. I'll just need each blob to record the nodes its visited, and then display those upon selection (and maintain the line even once the blob has been destroyed). Another is to create some sort of heatmap so that players can visualize what colors of blobs have been traveling through what highways in what quantities. That will be somewhat harder, though all I'll need to add in the core logic of the game are events. I can try color-coding parts of BlobHighway to make it more evident what blobs are going where, or at least make the permissions visible on the main map. I could also perform tracking from a specific society, making it so that when a player clicks on a society, they get lines tracing where the last X productions of that society have gone.
    b) All of these are fairly high-level UX tasks, many of which are likely beyond the scope of this sprint. But if the task of reasoning about the game become too difficult, these will be the first things I can use to make the difficulty more appealing.
12. The game doesn't describe the levels of complexity clearly enough, which might make it difficult for new players to understand what's going on and to properly plan for higher complexities.
    a) I'll need to add additional information in the Society pane, describing not just the same of a given level of complexity but also all of its relevant stats. Showing the complexity ladder itself might also be a useful endeavor.
13. Various UI issues might degrade the user experience so much that it becomes too difficult to reason about the mechanical challenges the game poses.
    a) The obvious solution to that problem is to resolve the UI issues. I've gone into some detail above about useful UX changes, and there are several known problems I need to resolve. Making the HighwayDisplaySummary more sensibly align its floating panes will help make permission control easier. Revealing edges on the map where highways can be placed will make reasoning about map topologies easier. Modifying the way in which Highways connect to a given MapNode will reduce highway overlap and prevent clutter. And changing the HighwayGhost length bug will make that system less wonky-looking. I might also change the way that ResourceBlobs are organized within BlobSites. Grouping them together

by type might be rather useful.

b) If, after all of those changes, the UI is still too unpleasant to engage with, I'll need to either attend to more sophisticated UI tasks (like those specified in 11), come up with something new, or rethink the underlying design.

Sprint Backlog

To Do

Critical (0 hours of work remaining)

1.

Important (0 hours of work remaining)

1.

Desirable (16 hours remaining)

1. Give players the ability to control ResourceDepot capacities (up to a point).
   a) Estimated 30 minutes to complete.
2. Build a paper prototype to experiment with alternate resources on the same horizontal level, and how those might combine to require more complex and challenging transportation networks.
   a) Estimated 8 hours to complete.
3. Create a system that allows players to select blobs and determine where they have been, to track the flow of certain resources throughout a given network.
   a) Estimated 2 hours to complete.
4. Add the ability to speed up. slow down, and pause the game.
   a) Estimated 30 minutes to complete.
5. Build a BlobAlignmentStrategy that keeps blobs of the same type in a cohesive group without causing unintelligible patterns of movement.
   a) Estimated 1 hour to complete.
6. Design and implement a system by which players can control the capacities of a Society.
   a) Estimated 3 hours to complete.
7. Design a UI that indicates on the world map whether a Society has its needs satisfied and how close it is to descending if its needs are unsatisfied.
   a) Estimated 1 hour to complete.
8. Fix some of the performance issues that seem to have arisen around the creation of BlobHighways and HighwayManagers.
   a) Estimated 2 hours to complete.

In Progress

Critical

1.

Important

1.

Desirable

1.

Completed

Critical

1. Give the player the ability to suppress a society's ascension.
    a) Estimated 30 minutes to complete.
    b) Took 30 minutes to complete.
2. Modify HighwaySummaryDisplay to place the endpoint displays in more sensible places, so it's very clear which panel controls which endpoint and so that they don't overlap.
    a) Estimated 1 hour to complete.
3. Assemble a collection of configurations, with the logic of Digital Prototype 2.2, that generate a stable city, or determine that a stable city is too difficult to achieve.
    a) Estimated 5 hours to complete.
    b) Took ~2 1/2 hours to complete.
4. Construct and execute some experiments that test what effects a given map topology has on one's ability to create stable configurations.
    a) Estimated 2 hours to complete.
    b) Took 2 1/2 hours to complete.
5. Implement the ability for players to select and destroy ResourceDepots in game.
    a) Estimated 2 hours to complete.
    b) Took ~ 1 1/2 to 2 hours to complete.
6. Change Society resource permissions so that they cannot import their productions.
    a) Estimated 30 minutes to complete.
    b) Took ~30 minutes to complete.
7. Experiment and record what effects forbidding Societies from importing their productions has caused.
    a) Estimated 2 hours to complete.
    b) Took 1 1/2 hours to complete.
8. Experiment with various need, want, and ascension paradigms for the four societies, with the goal of creating a more intuitive system that minimizes common pitfalls like accidental ascension or developing a level of complexity that the player has no way of supporting.
    a) Estimated 5 hours to complete.
    b) Took ~3 1/2 hours. I encountered a persistent bug that took some time to resolve, but also determined that a lot of the questions this task was asking weren't really relevant anymore.
9. Add the ability to upgrade a Highway multiple times.
    a) Estimated 3 hours to complete.
    b) Took ~1 hour to complete.
10. Design a UI that allows players to control Highway permissions more easily.
    a) Estimated 1 hour to complete.
    b) Took ~ 30 minutes to complete.
11. Establish and unit test the commands necessary to create and destroy HighwayManagers.
    a) Estimated 2 hours to complete.
    b) Took ~ 1 1/2 hours to complete.
12. Build the UI for creating, destroying, and observing HighwayManagers, and then debug the full integration.
    a) Estimated 2 hours to complete.
    b) Took ~ 1 1/2 hours to complete.
13. Create a module that can arbitrarily generate resources at arbitrary intervals of time.
    a) Estimated 1 hour to complete.
    b) Took 1 hour to complete.

14. Analyze the consequences HighwayManagers have on the execution of the game.
    a) Estimated 3 hours to complete.
    b) Took ~6 hours to complete, though the task was ill-defined and I terminated it rather arbitrarily.
15. Reorient the prototype to meet the specifications of experiment 7.
    a) Estimated 5 hours to complete.
    b) Took 4 1/2 hours to complete.

Important

1. Make MapEdges visible during play.
    a) Estimated 30 minutes to complete.
    b) Took 30 minutes to complete.
2. Give ResourceBlobBase the ability to clear its movement commands, and use that ability to improve Highway pulling and blob realignment to prevent senseless blob jitter.
    a) Estimated 1 hour to complete.
    b) Took ~5 minutes to complete.
3. Redesign the BlobDistributor module to improve the way in which resources are distributed based on priority, and to delegate resource pull rate to the Highways instead of to the BlobSites and BlobDistributor.
    a) Estimated 2 hours to complete.
    b) Took 4 hours to complete.
4. Resolve the bug that sometimes leads to the attempted destruction of a null ConstructionZone.
    a) Estimated 2 hours to complete.
    b) Took 12 minutes to complete.
5. Fix the length issues with HighwayGhost so that an unbounded drag operation no longer generates an absurdly long ghost.
    a) Estimated 1 hour to complete.
    b) Took 20 minutes to complete.
6. Fix highway creation so you can drag highways off of ResourceDepots.
    a) Estimated 1 hour to complete.
    b) Took >10 minutes to complete.
7. Fix ResourceDepot selection so that it doesn't select the underlying MapNode.
    a) estimated 30 minutes to complete.
    b) Took ~5 minutes to complete.
8. Determine how adding multiple Highway tiers has affected the game, and experiment with different compositions of tiers.
    a) Estimated 2 hours to complete.
    b) Took ~1 hour and 10 minutes to complete.
9. Establish the notion of a HighwayManager that's necessary to construct advanced highways and acts as the locus of highway maintenance.
    a) Estimated 5 hours to complete.
    b) Took 9 hours to complete (minus the UI and some of the SimulationControl commands, which were split off into a different task).
10. Fix the scaling issues for blobs contained within the BlobSites on MapEdges.
    a) Estimated 30 minutes to complete.
    b) Took ~30 to 45 minutes to complete.
11. Unify the common behaviours of the various panels into a single class called UIPanel.
    a) Estimated 1 hour to complete.
    b) Took 1 hour to complete.

12. Extend the Society display UI to include more detailed information on its current complexity, as well as its ascent and descent societies and the complexity ladder it's currently on.
    a) Estimated 2 hours to complete.
    b) Took ~2 hours to complete.
13. Add positioning logic to UIPanel to make sure that its bounds never leave the screen.
    a) Estimated 3 hours to complete.
    b) Took ~ 1 1/2 to 2 hours to complete.
14. Fix all the broken unit tests.
    a) Estimated 5 hours to complete.
    b) Took 1 1/2 hours to complete once the task came up, but a lot of the broken unit tests were resolved by other work, so it's hard to say for sure how much effort went into this task.

Desirable

1. Replace the standardized connection points on BlobSite with a generalized system that places Highway endpoints on a circle of a certain radius.
    a) Estimated 2 hours to complete.
    b) Took 30 to 40 minutes to complete.

Abandoned

Critical

1. Experiment with map configurations that involve high concentrations of ResourceDepots in order to determine if an overabundance of ResourceDepots leads to untenable problems.
   a) Estimated 3 hours to complete.
   b) Was deemed an ill-defined and meaningless task given the changes that had come before it.
2. Create some experiments that test how loosening and tightening the margins of error for various city-building configurations changes the map topologies that can be constructed.
   a) Estimated 2 hours to complete.
   b) I just don't think that I care about the results of this question right now. Having additional maps is nice, but I'm more interested in building the core mechanical loops in a way that's engaging on even a single map. Developing a theory of Strategy Blob mapbuilding is an important task for a later date.
3. Do some ad-hoc fiddling to see how the new, more expressive HighwayManager paradigm feels.
   a) Estimated 2 hours to complete.
   b) This task was almost tautologically poor. It serves no purpose, and so I abandoned it after remembering that.

Important

1. Fix the positional bugs involving BlobSites attached to MapEdges.
   a) Estimated 30 minutes to complete.
   b) This seems to have disappeared as a problem, but it's hard for say to sure if it's truly gone.

Desirable