**System Order**

***First, we must agree common uses based on good practices***

Starting with the use of classes and interfaces, abstract classes are used to not repeat code, since the basic functionalities are shared by several sets over several layers of complexities, as will be seen later, and of course concrete classes are used to depict objects in the game. Interfaces will be needed mainly to give an extra layer of categorization to the whole object systems, because of its size it’s required a particular denomination to each kind of object; and also, to give general functionalities but with substantially different implementations.

**The Four Main Structures**

***What is a Main Structure?***

The game is managed through four object structures, one to depict the “physical space”, the ground where people could be, another for people depiction called “population”, which is an interface and two literal objects inherit from it, “polis” for people and “citizen” for each individual. The next two main structures are somewhat more complex, because one is for the social activity depiction, including economic and politic movements, and it’s called “activity”. And the remaining is for “production” depiction, and it has every single good and service there are.

***How does the object abstraction work here?***

The objects are used, besides classify elemental objects of the game, they also could depict some of its properties, with a limited scope, and could weave its own system logic. So, consider the possibility of finding systems that escaped the logic of the main parent system, and regard them rather as sub-systems. Also regard that the quotation marks (“”) will be used in this document as a form to mark names that are literally the same in hard code, but wrote in narrative text format.

From now on, we are going to use the formal and literal names extracted from hard code, so please, note the capital use of words, they could be real code names rather than proper names. So, keep in mind that properties, methods, classes and interfaces use “PascalCase” naming convention, if literal object depiction is needed, then deem that these are named using “camelCase”. Just to know, none of the “kebab-case” or “snake\_case” were used.

**Spaces**

**IPhisicalSpace interface**

The interface for the base classes that depict the spaces that are occupy, the world itself. These classes are, from bigger to smaller in scale, Region (with a hexagon shape), Zone (triangle shape), Area (one of the sixth triangle portion shapes), and Parcel (with a square shape). Each one has a property with one or a set of the next “physical space” level of scale, and, of course, the “population” or society that you can control is individually in the smallest portion. Let's describe them one per one sorting them in the inversed order, this is, from smaller to bigger ones, after listing them.

* Land
* Parcel
* Area
* Zone
* Region
* Territory
* Continent

**Parcel class**

This is the minimum land unit that receives all the higher parameters in the world construction, where all the products converge (only the raw one at very beginning) and inhabitants (producing elaborated products just at start) start to settle. Your people begin in the center (the Land of the center), this means the Parcel has six blocks that can be used for building the minimum building space called “lands” (with the Land class), keep that in mind to understand the urbanization.

Now, within each Parcel could have different natural resources (raw products) depending to its “occupancy” and “terrain feature”, basically these are the characteristics that define the kind or type of land it is. In the next module we’ll see these classifications.

**IOccupancy interface**

To define the kind or type of land we must start defining a more abstract classification, which is a practical and about a global classification of the land, these are:

* SweetWater
* Saltwater
* FertileLand
* AridLand
* Woodland
* DenseForest
* Buildings

Each one has its single distribution of the resources, and it’s the first set of distribution, after this set up the process continue to the next stage that will map more the distribution, and closes it to the final configuration.

**ITerrainFeature interface**

Once the Parcel take the Occupancy classification, we must to define several factors that shape the infinite Parcel’s distribution change chain, because they are influence by a set of “features”, or constant factors, that are changed moving the absolute position, so each Parcel will have its single features due to its position in the entire map.

First, the Parcel retrieves from its parent container the set of ITerrainFeature already fixed by them (the Zone in the Parcel case), then the distribution is set up.

**Continent class**

Let's see, we already said the bigger classification is Continent, but the first parameter that it’s required to start the operation will not affect entirely to that land or space level, because is too big. This parameter is the ILatitude (class), once chosen, the continent will be placed depending to the distant from the equator.

**ILatitude interface**

The center of the continent will be exactly your starting position only in the case you choose Tropical as Latitude, therefore your position will be moved from the center of the Continent a 20% to the “pole”, whether you choose Subpolar or Subtropical as Latitude. And if you choose Polar as ILatitude the center of the Continent will be 35% of your starting position. This starting position will be moved later.

Let's listing them (with the available IBiome list after colon, sorted by its weighing):

* **Tropical**: TropicalRainforest, TropicalDryForest, GrassSavanna, SubtropicalRainForest, SubtropicalDryForest, TreeSavanna
* **Subtropical**: XericShrubland, TemperateForest, TemperateSteppe, GrassSavanna, AridDesert, SemiaridDesert, SubtropicalRainForest, SubtropicalDryForest, TreeSavanna
* **Subpolar**: TemperateForest, TemperateSteppe, DrySteppe, XericShrubland, AridDesert, SemiaridDesert
* **Polar**: Taiga, Tundra, TemperateForest, TemperateSteppe, DrySteppe

**IBiome interface**

Then we can get the “wind force” base, the “current force” base, and more ecological or weather features like the available IBiome list (listing them after the colon). This last one is exactly the next necessary parameter that follows. IBiome interface declare the main weather conditions (ITemperature and IHumidity), determine the altitude probability, and modify the previous features.

Here's a list of IBiome classes, from hot arid to cold arid:

* AridDesert
* XericShrubland
* SemiaridDesert
* GrassSavanna
* TreeSavanna
* TropicalRainforest
* TropicalDryForest
* SubtropicalRainForest
* SubtropicalDryForest
* TemperateForest
* TemperateSteppe
* DrySteppe
* Taiga
* Tundra

Each one has a range per property feature, setting boundaries to the rest of climate variables, which could be a lot (scilicet, Precipitation, WindForce, CurrentForce, AtmosphericPressure, Vegetation, ReliefFactor, and Continentality), but the remaining main ones are the set by IAltitude, ITemperature and IHumidity interfaces, where IAltitude is the last one to be set up.

**ITemperature interface**

The next is ITemperature, because with the ILatitude a scaled list of ITemperature values can be set, and then with the IBiome scattered to the game map the ITemperature object properties can be modified, acting like a conditioner feature.

This is a list of ITemperature from hotter to colder:

* Suffocating
* Stifling
* Hot
* Warm
* Templated
* Cool
* Cold
* Chilly
* Frosty

And each one is a temperature range which current value is updated in each game turn, and this modification depend now (within the previous greater boundaries set up) to the time and the counter-influence that the IHumidity could be sent.

**IHumidity interface**

This last feature is set according to several factors. First, and main is the self ITemperature, more precisely the current Grade property, setting the “maximum steam capacity” each game turn, key factor to determine future Precipitation times set up.

IHumidity are classified in the following classes, from the driest to the wettest:

* Barren
* Damp
* Dank
* Dry
* Humid
* Moist
* Parched
* Withered

**ITectonicPlate interface**

Now we can begin to drawing the first “relief” lines, literally, but for that we need the third factor that is needed mandatorily to be chosen, the ITectonicPlate, this will determine the possibilities for the relief forming; regardless of final player position, these ITectonicPlate will be set to stablish a “finite world”, or literally the world.

The option is according to the size basically, and there aren’t many to choose from:

* BigContinentalPlate
* SmallContinentalPlate
* SmallMaritimePlate

Obviously, the most common ITectonicPlate that will be played is BigContinentalPlate, this is composed by giant extensions with or without big water masses, but definitively with a lot of seas and lakes, with river connections. But first comes the IAltitude distribution, especially the higher ones, for this the “finite world” will be set up only with the main ITectonicPlate and their neighborhoods. With this we can set up the direction of the Displacement property, regarding some rules.

**Distribution of the Existing “Finite World” Tectonic Plates**

The probabilities of having neighbors are as follows:

. 100% to have at least one BigMaritimePlate (yes, it exist but you can’t use it), and one BigContinentalPlate as neighbors, but the remaining space is for sale!

. 50% to have only a third remaining neighbor, which, within, a 35% (from total) could be a BigContinentalPlate, and the 15% to BigContinentalPlate.

. 30% to have two more remaining neighbors, 10% to have one BigContinentalPlate and one SmallContinentalPlate, 10% to have one BigMaritimePlate and one SmallContinentalPlate, 8% to have one BigContinentalPlate and one SmallMaritimePlate, and 2% to have one Big ContinentalPlate and one BigMaritimePlate.

. 20% to have three more remaining neighbors, 8% to have 1 BigContinentalPlate, 1 SmallMaritimePlate and 1 SmallContinentalPlate, 6% to have 1 BigContinentalPlate and 2 SmallContinentalPlate, 4% to have 2 BigContinentalPlate and 1 SmallMaritimePlate, and 2% to have 2 SmallContinentalPlate and 1 SmallMaritimePlate.

. 8% to have four more and 2% to five more, in these cases each slot must be drawn from scratch with the following percentages: 35% to BigContinentalPlate, 20% to BigMaritimePlate and SmallContinentalPlate, and 20% to SmallMaritimePlate.

**MovementType property; the Tectonic Plate Direction**

Once set the distribution, to set the ITectonicPlate directions, we have three MovementType properties to construct each one of the boundaries between plates; these are, convergent ones for the plate's collision, divergent ones for plates division, and the transform boundaries for plates lateral displacements. So, the rule leads; first, the northernmost ITectonicPlate, when the chosen ILatitude is Polar, has always divergent movements, otherwise, this “movement direction” has independence of the rest, it depends on its own boundaries/borders. But before dive into this aspect, let’s review what these three “movement types” are.

MovementType property possible values for ITectonicPlate options:

. Convergent: is where the oldest plate suffers a “subduction” displacement below the newest ITectonicPlate in a collision between them, one side will have a “trenched” zone, and the other will have a “mountain range” as result.

. Divergent: is where a “rift valley” zone is generated, in continental boundaries (continental plate against another one like it), or a “sub-maritime mountain range” in maritime boundaries, due to a division in their displacements.

. TrasformBoundary: is where a “faulting” zone is created, whether sub-maritime or not (in any kind of tectonic plate), due to lateral plates displacement.

**The Tectonic Plate Connection Net**

Now, several “magma activity hubs” will be set, not many really, among the wide “finite world” created, exactly in the divergent boundaries, where this “magma hub” would apply a hypothetical force that created that kind of plate movement. This is important to know because on the opposite side convergent movements will be applied, and in the lateral ones a transform boundary will be applied.

So, yes, we could say that divergent movements are the first ones to be set up. Consequently, convergent and transform boundary sides are automatically deducted.

**PlateType; the Tectonic Plate Constitution**

Now we got to use a property to define the entire space kind, this is PlateType property, which values could be:

. Shield for ancient and stable plates.

. Massif for not so ancient and stable plates.

. MountainRange for relatively new and unstable plates.

Shields are set in relatively big areas, massifs are next to mountain ranges, these last ones are found above the boundaries only, but shields also could occupy those zones (although also have high chances to be in the middle of the ITectonicPlate space). But all these calculations must be done further. First, we must deem that in maritime against maritime plates, the boundaries are filled with mountain ranges made of “sub-maritime mountain range”,

Of course, the big line of “mountain ranges” will be the origin to the higher lands, and they will be forming gradually, in a decreasing exponential curve to zones more plains or until to reach the coast (where the altitude becomes negative).

The Continent will be placed in the plate, along with their neighbors, and the plates will fill their ground with the PlateType, MountainRange will be in at least one side of the ITectonicPlate frontier, but if the continental ground isn’t over a transform boundary border, then the ground is a MountainRange PlateType kind, if it is over a transform boundary border, then it a half of the chance to be a Shield PlateType kind, unless a MountainRange comes below, so to speak, forming a long mountain chain.

Shield are mainly set over the center of the continental ground, and Massif depends entirely on the MountainRange. It's clear that for maritime plates it only uses OceanFloor grounds.

Finally, the tectonic plates are set up, then marginal weather factors can be set, like final WindForce adjustments, temperature modifications and consequently the corresponding humidity changes. With these final values, the resources over each land square, set them up in first stages, can be modified by this new atmospheric configuration. This is how the physical spaces are set up without water specifications, in a very rough way.

**Products**

**IProduct interface**

The products supposed to be every usable unit inside the “physic space” or land, including the result of the natural or artificial “production”, including their waste. First, we got the “raw products” that depends directly to the land type combination (set up in the previous module above); second, we got the manufacturing products, and there're several layers in the production chain, all the product transformations take goods from this product kind. Last, we got the “final products” that comes to be the services or consumer goods.

Every “product” has its own common properties; all of them have at least one ISocietySector interface associated, this is to relate all the production chain where that single product is being transformed to the consume phase (which in the early time some would be directly consumed from raw stages), but it's needed to make this relationship in a very wide scope.

Another common property is the so-called Classification, these are more detailed categories for the “society sectors”, and are the final definition that segregates the activities in your society; they could have similar functionalities with others classifications but clearly are of different nature, and required different professionals to make them.

But all of above have plenty to do with the further Social Activity module, please, regard this to find these interfaces in the Activity group in the Models root (into the hard code). Instead, the Type property describes what the “product” actually is, and what a worker can do with it. It is the material condition that enables some specific functionalities.

But first we are going to give a gaze to each one of these properties, remembering that the first two comes from the Activity module, so to find them, check in Models/Activity/Interfaces hard code root.

**ISocietySector interface**

A “society sector” is any category in your society that should be taken with an independent gaze. They have their own particularities and rules inside the bicephalic society control (I refer to Politic and Economic Knob, see this further, in Socety Activity module or in the Main Services document). That's because of why some economic sectors are agglutinated, and others seem to be separated when they could be just one sector. This is the rare case of IFinancialSector and ICommerceSector class, at the end both are related to buy and sell things, but they have particularities that are needed to be deemed. Generally, the case is exactly the opposite, ISocialSector have services so different in nature like “products” of Medical, Religiosity or Cultural “classifications”, IKnowledgeSector have Education and Sciencie “classifications”, and the classic IPoliticalSector have Politic, Legislation, Policy and Army “classificatins”, so keep that in mind. One leads the functionality; another leads the working nature or essence.

The list of **ISocietySector**:

* IFoodSector
* IGarmentSector
* IConstructionSector
* ICommercialSector
* IFinancialSector
* IIndustrialSector
* IWasteSector
* IEnergySector
* IRentalSector
* IPoliticalSector
* IKnowledgeSector
* ISocialSector
* IChemicalSector
* ITraffic

**ISectorClassification interface**

Managing “sectors” is ok to give this game project a path where the economy comes first, then the politics can be dived in, but yet we don’t need to focus on cultural or deep social parameters like sociologic, at least for now...

That's why ISectorClassification interface is needed, to classify the real economic/politic sector, and give them their particularities and nuanced parameter.

These are the ISectorClassification interfaces for the **IRawProduct** ones:

* IMaritime
* ILivestock
* IWildAnimal
* IAgriculture
* IVegetation
* IResidue
* IRawMineral
* IRawMetal

ISectorClassification for **IManufacturedProduct**:

* IFood
* IFeeding
* IPower
* ITransportation
* IVehicle
* IRefinery
* IWood
* IMetal
* IIndustrial
* IJewerly
* IMachine
* IChemical
* IConstruction
* IBuilding
* IFabric
* IClothing
* ICraft
* IManufactured
* IResidue
* ITransport
* ITraffic

ISectorClassification for **IFinalProduct**:

* ICulinary
* ICommercial
* IScience
* ISecurity
* IMedical
* IEducation
* IFinancial
* IPolitic
* IArmy
* IService
* ILegislation
* ICultural
* IPolicy
* IReligiosity

**Product Type interfaces kinds**

We don’t got something like IProductType, instead we call “product type” or Type to all the interfaces that inherit from the production chain origin, these are, IRawProduct, IManufacturedProduct, IFinalProduct interfaces, and their got the special material conditions related to its creation sources.

These are, from the **IRawProduct** list:

* IExtractable
* IEdible
* ISpinnable
* ILively

From the **IManufacturedProduct** list:

* ITransportation
* ITechnological
* IFabricable
* IWeavable
* IRefined
* IPerishable
* ITransportation
* IEnergy
* IBuildable
* IImprover
* IDrug
* IChemical
* IPurified
* IResearchable
* IProcessed

From the **IFinalProduct** list:

* IArmedWing
* IPolicy
* IPolitical
* ICultural
* IProtector
* ITransit
* IFinancial
* IDisposable
* IServiceable
* IServeable
* ISupplyable
* IInterchangeable
* IBelievable
* ILearnable
* IReasonable

**IRawProduct interface**

All the units created in the nature are considered “raw products”, the so-called resources, so the configuration described in the module above set up all the environment settings that sets the “raw products” distribution, distribution that ends up proportionally packed in each one of the nine Land spaces within the Parcel locations.

When the population begins in a Parcel, they automatically begin to expand the “known space” looking for this kind of products, moving throughout the “parcels” ‘till find good goods to produce what they need (explained later in Social Activity module).

Basically, they depend on the ITerrainFeature of the IPhysicalSpace object occupied, then a set of logical operations are made to define their actual locations, the Type set the condition of the “resources grabbing”, and consequently, the “resources consumption”; the Sector parameter gives the production chain line, and the Classification gives the particular features and complexity.

The IRawProduct, as the rest of the IProduct children, their final object classes use at least three kinds of interfaces, the one that inherits, I mean this one, the Product Type, and the rest come from the Activity module, but not inherit from IActivity, because they’re just categorized based on their productive chain place, they are not a specific interface type for the Activity module but all modules in general, the Activity module and Production module mainly, and the Population module also would have to appeal to these ISocietySector and ISectorClassification interfaces. So, please regard this, the Product Type are the one interface set that the object classes of products inherit from; and the Product Classification or Product Sector are global interface sets for several game modules, but directly related to Activity module.

Understanding this, here’s the IRawProduct classes classified or divided by ISectorClassification interfaces:

**Maritime**

⦁ running water (sweet water)

⦁ hydraulic force (sweet water - saltwater - building)

⦁ fish (saltwater)

**Livestock**

⦁ beef cattle (fertile land)

⦁ sheep cattle (fertile land)

⦁ pig cattle (fertile land)

⦁ equine cattle (fertile land)

**Wild Animals**

⦁ wild cervids (fertile land - woodland)

⦁ wild pigs (fertile land - woodland)

⦁ wild bovines (fertile land)

⦁ wild equines (fertile land - woodland)

**Agriculture**

⦁ grains (fertile land)

⦁ vegetables (fertile land)

⦁ tubercles (fertile land)

⦁ fruits (fertile land - woodland)

**Residues**

⦁ wastewater (building)

⦁ scraps (building)

⦁ industrial waste (building)

⦁ leftovers (building)

**Minerals**

⦁ clay (fertile land - arid land)

⦁ decorative rocks (arid land)

⦁ salt (arid land)

⦁ solid rocks (arid land)

⦁ coal (arid land)

⦁ fusel oil (arid land)

⦁ gas (arid land)

⦁ rare minerals (arid land)

⦁ precious minerals (arid land)

**Metals**

⦁ metalloids (arid land)

⦁ iron (arid land)

⦁ aluminum (arid land)

⦁ copper (arid land)

⦁ precious metals (arid land)

**Vegetation**

⦁ hard woods (woodland - dense forest)

⦁ soft woods (woodland - dense forest)

⦁ culinary herbs (fertile land - woodland)

⦁ medicinal herbs (fertile land - woodland - dense forest)

⦁ rubber (woodland - dense forest)

**IManufacturedProduct interface**

Technically everything extracted in the first stage could be consumed immediately (following of course the “marginal utility increasing curve”, explained further in Population module), but if the “Citizen’s Capacity” is left after filling all the prioritized “necessities”, then he became to invest that energy in enhance its job, and if he got the Capital enough, he can Invest to solve it somehow, because the Capital used could be not so “liquid”.

But if so, then your society begin to produce IManufacturedProducts, at first of course only the most primitive products but with the time better and better products should output. And only they will be produced if the entire consumption chain is made, and this often starts from the final stage, in the opposite direction, because that products aren’t consumed in this “manufactured stage”, and only exist by a service proposal.

Manufacturing products list:

**Food**

⦁ drinking water

⦁ meats

⦁ salt

⦁ seeds

⦁ vegetables

⦁ alcoholics

⦁ non-alcoholics

**Feeding**

⦁ storage

⦁ fridging

⦁ ferment

**Energy**

⦁ coal

⦁ mechanical force

⦁ electricity

⦁ gas

⦁ fuel

**Transportation**

⦁ heavy transport

⦁ heavy shipping

**Vehicles**

⦁ boat

⦁ ship

⦁ car

⦁ cart

⦁ plane

⦁ train

⦁ truck

**Refinery**

⦁ purified

⦁ cast

**Wood**

⦁ cut wood

⦁ dried wood

**Metal**

⦁ iron

⦁ steel

⦁ aluminum

⦁ copper

⦁ metalloids

**Industrial**

⦁ air conditioner

⦁ compressor

⦁ crawler

⦁ electronic devices

⦁ fridge

⦁ furnace

⦁ gears

⦁ gigant tanks

⦁ kiln

⦁ lathe

⦁ rotors

**Machine**

⦁ bulldozer

⦁ crane

⦁ digger

⦁ drill machine

⦁ feller

⦁ lifts

⦁ sewist

⦁ tractors

⦁ welder machine

**Jewelry**

⦁ precious metals

⦁ precious minerals

⦁ glass

**Chemical**

⦁ acids

⦁ preservatives

⦁ fertilizers

⦁ material protectors

⦁ drugs

⦁ bactericides

⦁ dye

**Constructional**

⦁ rocks

⦁ decorative rocks

⦁ concrete

**Building**

⦁ started

⦁ structural

⦁ built

**Fabrics**

⦁ latex

⦁ fabrics

⦁ leathers

**Clothing**

⦁ cheap clothes

⦁ expensive clothes

⦁ luxury clothes

⦁ cheap shoes

⦁ expensive shoes

⦁ luxury shoes

**Crafts**

⦁ ornaments

⦁ crafting

**Manufactured**

⦁ appliances

⦁ art tools

⦁ construction tools

⦁ crafting tools

⦁ cutlery

⦁ fabrication tools

⦁ furnitures

⦁ textile tools

⦁ weapons

**IFinalProduct interface**

Practically, the final stage are entirely services, and the goods that aren’t then are service’s goods, or good that only are used by those services (like financial assets, transport, laws and decrees). They are produced when the market balance set one side with good basic economy competitive and performance, but with no essential services required, and another side with no so good competitive performance in basic sectors, but they could supply that services demand, especially if they are qualified to do so.

Final products list:

**Medical**

⦁ medical attention

⦁ medicaments

⦁ hospitalization

**Traffic**

⦁ Dealership's

⦁ Fleet

⦁ Squadron

**Transportation**

⦁ business transport

⦁ public transport

⦁ civil transport

**Legislation**

⦁ precarious

⦁ primitive

⦁ deductive

⦁ utilitarian

⦁ excessive

**Commerce**

⦁ appliance's

⦁ bazaar's

⦁ books

⦁ cleaning's

⦁ furniture's

⦁ hardware's

⦁ jewelry's

⦁ real estate

**Policies**

⦁ policy

⦁ politics

**Financial**

⦁ stocks

⦁ assets

⦁ tokens

⦁ derivatives

**Religiosity**

⦁ assortment

⦁ religiosity

**Security**

⦁ scattered

⦁ nucleated

⦁ presented

⦁ controlling

**Knowledge**

⦁ basis

⦁ elaborated knowledge

⦁ sophisticated knowledge

**Education**

⦁ primitive

⦁ laborious

⦁ proffesional

⦁ thorough

**Cultural**

⦁ folk

⦁ popular

⦁ elaborated

⦁ vanguardist

**Service**

⦁ traffic

⦁ private cleaning

⦁ public cleaning

⦁ clerkship

⦁ assistance

**Culinary**

⦁ eatery

⦁ beverage

**Army**

⦁ infantery

⦁ cavarly

⦁ navy

⦁ artillery

⦁ air force

All the products must be created and held in specific Occupancy Parcels, and of course occupy a space in their corresponding slot, according to their nature and magnitude.

**Population**

**IPopulation interface**

The inhabitant units, called Citizen, have “psychological parameters”, that impacts over the “economical ones”, so this is a “demographical” and “sociological feature” that increase significantly the complexity even in the initial version, which excludes deep parameters like pure exclusive psychological parameters, only shallow ones, or sociological ones.

To be clear, the “sociological features” are for now the Class, somewhat different of the classic concept of a social class, but subtle. We need to include the age and sex because of their importance of the economic growth, at least from a historic point of view, in the sex case, but specially because the need to simulate the births, but the complexity seen is too costly to be afforded for now.

So, in the “demographical features” we got only the Status (what is usually called a social class) and the Niche properties, where Status comes up as the purchasing power of the individual, and the Niche, of course, its profession. Status has somewhat fixed ranges that determine their limits among classes, and what it handles is mainly the consumption/investment behavior (understanding investment as Capacity application literally). And Niche is achieved through the practical working, that starts from 0 Efficiency, and increase with the Experience accumulator, but there’re some niches that require some levels of first and cheaper niches.

**IDemographyFactor interface**

With “demography factors” we refer to the hard population parameters, such as Reproduction, Age, Sex (both last not considered in first launch), Wealth, Profession, Health, Education, Knowledge, Criminality, and so on.

**IStatus**

* Rich
* Wealthy
* Professional
* Poor
* Pauper

**INiche**

**IFoodSector**

* Breeder
* Farmer
* Fisher
* Gatherer
* Hunter
* WaterSupplier
* AbattoirWorker
* Butcher
* Distiller
* Bartender
* Cook

**IGarmentSector**

* Weaver
* TextileWorker
* Shoemaker
* Tailor

**IConstructionSector + IIndustrialSector**

* Miner
* Woodsman
* Lumberman
* Engineer
* Smith
* Carpenter
* Manufacturer
* Industrial
* Jeweler
* Mechanic
* Builder
* Architect
* Technician

**IEnergySector**

* Miller
* EnergySupplier
* Electricist
* GasFitter

**ICommercialSector**

* Merchant
* Storekeeper

**IFinancialSector**

* Trader
* Banker

**IRentalSector**

* Holder

**IWasteSector**

* WasteCollector

**ITransportSector**

* Rider
* Chauffeur
* Marine
* Pilot

**IPoliticalSector**

* Chevalier
* Politic
* Police
* Soldier

**IChemicalSector**

* Chemist
* Pharmacist

**IKnowledgeSector**

* Teacher (IEducationCategory)
* Doctor (IHealthCategory)
* Priest (IReligiousCategory)
* Judge (ILegislationCategory)
* Lawyer (ILegislationCategory)
* Researcher (IScientificCategory)
* Theorist (IScientificCategory)

**ISocialSector**

* Artist (ICulturalCategory)
* Guard (ISecurityCategory)
* Artist (ICulturalCategory)
* Guard (IServicesCategory)
* Artist (IServicesCategory)
* Guard (IServicesCategory)
* Plumber (IServicesCategory)

**ISociologicFactor interface**

These are the subtle population parameters, or those who have their origin in mental activity. Some could be for example, Psychology, Ideology (these last are far away from being deemed to the first game versions), Aggressiveness, Tolerance, Sociability, Superstition, Frivolity, or Class, among others.

**IClass interface**

Class establishes the pattern to contribute to the economic/social production circle, and it is related to the “production chain” from the “social activity” module. These are, Worker to who apply his own work force to produce value for the market; Manager to those who pay each time some kind of premium to get the benefits from the settlement of a business; Owner to those who offer any kind of building or establishment and charge a rent for it; and finally, Capitalist to who lend capital in exchange for interest.

This set the production chain if you watch. Workers produce the coordinated productive work thanks Managers, who borrow loans from Capitalists, and rent buildings from Owner. Managers have good ideas but not have money to make them, Capitalists have money but no good idea, Workers have anything, and none have a place where to live, and here comes the Owners.

IClass list:

* Worker
* Manager
* Capitalist
* Owner

**Citizen properties**

All above is about properties, yes, but they are shared both for Polis and for Citizen object classes, but also both have independent parameter to measure. Citizen of course has individual properties that analyze their social decision based in their particularities, both inner and outer.

**Necessity-Capacity classes**

Much of the properties in each Citizen are the counter-part of another, the first of them are the pair of properties Necessity-Capacity that defines the motion motor of the character so to speak. A Necessity could be satisfied by a lot of similar or even different products, but always must count with a slot in Capacity; let’s say that Capacity is a number that must be used to fill several holes, the necessities that must be prioritize mandatorily to avoid experience a decrease in their Health. Capacity is a number that "recovers” each turn with a portion of itself depending the proportion left in the previous turn and the Health that he has, Necessity does something similar but in the opposite direction, the number decrease proportionally to the portion left in previous turn and it’s Health.

**Health class**

The more Health your Citizen has the less the Necessity decrease and the more Capacity will regenerate to be used the next turn. Health improves with a variety in the Feeding, but mainly with Hygiene and Sanitation in case of Sickness, also Fitness could help but it needs to develop “knowledge”. One of the more important factors is the Recess time, you can sacrifice some portion of the Recess time to obtain a plus on Capacity but it hits very strong to this factor, resulting in a later lack of “recessing time” and a reduced Capacity recovery.

**Recess-Recovery classes**

Keeping Recess factor normal will keep the regular proportion of Recovery, and this last factor determines the Capacity regeneration, or the Necessity satisfaction. So, Recess impacts hard on Health and proportionally on Recovery. Health is an independent variable that it’s influenced by several outer variables, Recovery is the pair property of Recess, it works with a proportion.

**Capital-Consume classes**

When a Citizen works (as a Worker or not) he creates Capital, Capital is all the Products that you are collecting when producing. At the end of the working day the Consume will use all these IProduct list that can be consumed to satisfy all its Necessity prioritized list. The leftovers are Capital available to be exchanged for other goods that are useful for your Citizen proposals. In a few turns some kind of good will be highly valued and becomes as exchange good, depicting first stages of liquid Capital, or money namely. So, one takes away from the other, and vice versa.

**Investment-Thriftiness Class**

You can do an effort to thrift voluntary, sacrificing always something for that, because if not the common action that a Citizen automatically do with its Capital no consumed is Invest it to become a Capitalist or a Manager (he only can be Owner if he Invest in Real Estate business), and takes advantage of that Thift, if not he can await keeping the Capital in their Saving Funds in a frugality act, and accumulate to do a bigger Investment.

This last scenario is the usual base for thriving and increase exponentially the total incomes through new production goods, called “working intense stage” in the production chain context, keep them away from consumption goods, or “capital intense stages”.

**Productivity-Charge**

All the economies are intrinsically competitive, how’s this? Because if they aren’t too productive, they could be cheaper, the people could be “poor” but they will not be “vacant” by self-initiative. So, without any IRegulation that forbids it, the cost decreases till find the balance in the Market.

**Labor-Laziness**

The Citizen that has this ratio Laziness leaned, will prefer automatically to increase its idleness to the detriment of the Time and Capacity used to satisfy their Necessity, the idleness becomes a Necessity itself.

**Social Activity**

**IActivity interface**

And finally, we got here, we already know about goods and services, and also about the individuals and the society itself. Now how these both things relate each other? Through some IActivity interface object, and it’s not only an “economic activity”, also could be a “political activity”, because we got two main control knobs just to speak; we can control the Economical Knob or the Political Knob, one with a market logic and the other with a strength logic, and once this parameter is set up, then the logic of power comes up with this. So, we are going to explain well this nuance before continue with the several types of IActivity interfaces.

**Economic Knob**

We can get a Citizen an order him a logical action, if not, he only behaves accordingly to his own interest, whether economic or political motion. But if the action has to be with something useful for him or anyone that enters to the Market, then the Economic Knob comes active. This is for control a set of options and functionalities that have a unique logic. The logic here is simple; with Necessity and Capacity set, the Capital is filled with the surplus, this can be offered, as his own work force to a Manager, then a Market is settled in some Location (Land influence to nearby lands) getting all the IProduct offered by Classification and Type, and there’re the offer, or the Supply as we call it, and then the demand is called, the Ask object class, and a Price is reached, giving place to transactions and information to further economic analyzation.

**Political Knob**

We can no go so easy here, with the Political Knob, because it’s moved by machiavellian reasons, and this includes lies and manipulation. But this knob is not able at the beginning, because it must be formed, it always occurs with a conflict event, but almost always because of a swollen Aggressiveness, when there’s enough criminals to engage in a criminal contract to subject the population, after this event the State is formed and it can be improved to not be a parasite that only offers the monopoly of Defense against other communities (in early game could be no communities rendered yet, but commonly they should be rendered, but no necessarily aggressive or conflictive against your society), if not others complementary.

But yes, the State is view here as a parasite to the Economy for a simple reason: in the Economic Knub the “wealth” is built by producing something asked by another person that agrees to buy it to him. But here the source of wealth is not asked by any Market, it is charge to anyone that can be charged against his willpower, and only through that source, and then the booty is used for whatever the Politic deems appropriate to keep his job (of course this could include the quench of the raged mobs, and plunge the riots, giving them special and costly favors).

This structures, as the economic one, evolves until reach a State where tries to offer goods but specially services, that the same mechanism that were described above will manage to satisfy, or at least create the delusion to help the society welfare.

**IActivity Stages, Sectors and Categories interfaces**

Having said that, IActivity is an interface to integrate all those options, but it has a lot of branches, these start with the “stage” that are basically three IActivity interfaces; IRoughActivityStage interface for rough and first “stage” that works with IRawProduct interfaces to yield IManufacturedProduct interfaces, IFabricationActivityStage for intermediary “stage” that works with IManufacturedProduct to yield another IManufacturedProduct or an IFinalProduct, and IServiceActivityStage to final “stage” and works with IManufacturedProduct to yield IFinalProduct, or work only with IFinalProduct in some rare cases like ICultural classification object interface.

Then we got the Sector class but not the ISocietySector interface, remember that it is used for production chain segregation goals, this Sector is not literally a class but rather an interface kind of IActivity grandchildren, like in products types, in activity all the interfaces used inherits from one parent, IActivity, then the three above, and now these Sectors; but there’re several, here’s how after this categorization comes the Categories, So we got that for Activity we got Stages, that they have Sectors which in turn have Categories. Now this is that list with Categories within each Sector item (after the colon):

**IRoughActivityStage**

* **IHusbandrySector**: IFarmingCategory (for eatable vegetations), ILivestockCategory (for general animals, including fishing).
* **IExtractiveSector**: IFellingCategory, IMaritimeCategory, IMiningCategory, IExtractionCategory.

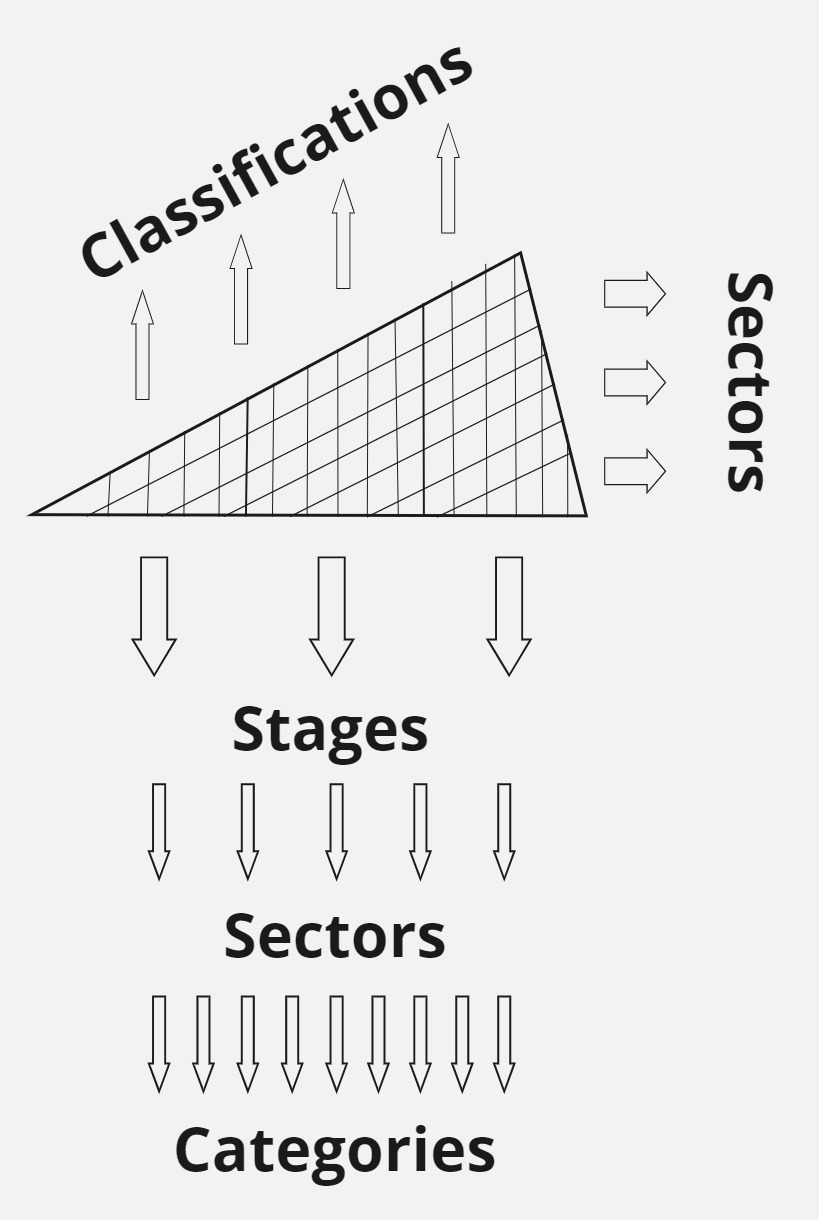
**IFabricationActivityStage**

* **IRefinementSector**: IProcessingCategory, IRefineryCategory.
* **IManufacturingSector**: IIndustrialCategory, IManufacturingCategory, ITechnologyCategory, INourishmentCategory, IPharmaceuticalCategory, ILaboratoryCategory.
* **IClothingSector**: ISpunCategory, IClothingCategory.
* **IBuildingSector**: IBuildingCategory.
* **ITransportationSector**: IWasteCategory, ITransportationCategory.

**IServiceActivityStage**

* **IBasicSector**: ILegislationCategory, IRegulatoryCategory, IReligiousCategory, IScientificCategory, IHealthCategory, IEducationCategory.
* **ICulturalSector**: ICulturalCategory, IKnowledgeCategory, ISecurityCategory.
* **IExchangeSector**: IMarket, IFinancialCategory.
* **IRecreationalSector**: ICulturalCategory.
* **ISocietyServiceSector**: ISupplierCategory, IGastronomyCategory, IServicesCategory.
* **ITrafficSector**: ITrafficCategory.
* **ILeasingIndustry**: IRentalCategory.
* **IDiplomaticSector**: IRegulatoryCategory, IPoliticCategory.

**ISocietySector and ISectorClassification interfaces**

I insist, don’t confuse yourself with the previous Sector definition, there’re two kind of different production chain abstractions, while the Sector in the IActivity interfaces line is a collective way to group several economic categories in just one, so they go from the three stages, and then the structure begins to display itself branching out in sector and categories (as leaves). But if you look at the picture here by the side, the production chain depicted as the triangle, had the three stages diving vertically but transversally the triangle is also split, these are the Sectors previously seen, as properties interfaces for the Activity object, because one thing is an interface that is for the object inheritance and another is for its property type. As you can see, Classifications are specific type that leads the way to work with category interface objects; and Sectors are like sector but that marks a transversal groove, not vertical. 

Once this is clarified, let’s unify everything in a single system or economic environment.

**Idleness-Yield**

The relation between the productivity of the Capital, and the leisure or repose of available Capital. This could occur in business where the high amount of Capital can’t be use because of payment imbalance that leads to unemployment and the no use of machines or Capital; it’d be more expensive produce that liquidate the production (or Yield) in the Market. Yield is that counterpart, is the use of Capital, the more Yield you achieve the more useful Capital your society benefits from.