Chaos or Order功能需求说明

Chaos or Order(以下简称Co2)是一款跨平台的集换式卡牌电子游戏，玩家可以在游戏中使用棋子，骰子，卡片来进行战斗，并使用不同的策略去达成任务目标。

Co2的核心是开源的，允许二次开发的，他允许没有安装jvm的人也能编辑部分游戏内系统(类似minecraft 的 mod)。

本文中的部分，主要描述的是Co2的服务端，即能够提供多房间游戏业务，抽卡业务，自定义卡片效果业务，处理战斗或非战斗操作具体逻辑的后端。

Co2系统,根据系统层级,主要分为三层:Prototype(原型层),Entry(实体层), Instance(实例层)，分别对应游戏的三个层级:数据存储，战斗外收集养成和战斗。

Co2系统内，包括以下独立的子系统:AstTree系统, Effect系统,骰子系统, 棋子系统，地图系统，卡片系统，主要难点集中在原型部分。

## 一、Scene

这个类别象征的是游戏开始后的”桌面”实体，也可以称之为”场景”，这个实体存在于GameSession(游戏会话)实体中，内部存储有玩家实体，卡牌实体，区域实体，注册器，事件记录等。当他结束的时候，本次游戏结束。游戏进程将会被封入GameSession中。

内部包含:

Registry : UniqueIdRegistry 实体的注册器

hasEnded : Boolean 本场景是否已经结束

isClosed : Boolean 本场景是否已经关闭(暂停)

turns : Turns 回合实体(暂未完成)

zones : Map<PlayerInstance,ZoneInstanceSet> 玩家及其区域的映射，key为玩家实体(见下文)，value为玩家使用区域的集合(见下文)。

venueMap : VenueMap 游戏时使用的场地地图。(见下文)

## 二、Card-CardEntry-CardInstance

这三个类别代表的是游戏时使用的卡片的三个形态:在数据库中的原型，在玩家收藏中的实体，以及具体使用时的实例。三层互相隔离。

### Card

卡片的原型，密封类，具有泛型<E : Effect>(Effect见下文)，

### AstTree(抽象语法树)系统

AstTree(抽象语法树)系统，用于将Function方法进行序列化。这个系统可以让某些实体的特殊效果作为json被序列化,从而达到可以将卡片保存在数据库中，并且让二次开发者可以通过直接操作数据库中存储的json\_来修改卡片的效果。

该系统除了主系统，还包括两个小系统:

Param(传参)系统，仅供为AstTree\_系统服务，将所有AstTree运行时需要调用的传参打包统一成object,这些传参将会被astTree的末端paramLeaf 节点捕捉到，并强制转换咸对应类型的对应参数。除此之外，还包含一个Situation指针，以供处理具体逻辑。

Situation(情景指针)系统，用于存储效果处理前/效果发动时/效果处理中/效果处理后才存在的一些instance对象，以供所有具有特殊效果的实体的状态检查和具体逻辑操作。内部包含:操作使用的scene(情景，也可以称之为游戏房间)、player(使用效果的玩家实例),agent(效果的载体),initiator(使用效果的棋子实例),target(效果对象),input(效果确定发动后使用的传参)，以及更多。

注:因为效果处理每个阶段所需的对象不同，有些对象可能在指定时点时并不存在，所以situation需要根据时点进行细化。

二、Effect(实体效果)系统

Effect(实体效果)系统，整个游戏的核心。负责存储指示物实例(由卡片效果生成)/ 棋子实例/卡片实例的效果。该系统包括:

1.处理前的效果检查，在一个操作结束后处理，和游戏失败的检查处于同一