软件架构大作业要求和分析

背景

需要关注的地方，加粗显示

信息技术与经济社会的交汇融合引发了数据迅猛增长，数据已成为国家基础性战略资源，大数据正日益对全球生产、流通、分配、消费活动以及经济运行机制、社会生活方式和国家治理能力产生重要影响。2015年，国务院印发了《促进大数据发展行动纲要》，纲要中提到，数据已成为国家基础性战略资源。目前，我国在大数据发展和应用方面已具备一定基础，拥有市场优势和发展潜力，但也存在政府数据开放共享不足、产业基础薄弱、缺乏顶层设计和统筹规划、法律法规建设滞后、创新应用领域不广等问题，亟待解决。

为贯彻落实党中央、国务院决策部署，全面推进A市政府的大数据发展和应用，提升政府服务和监管能力，A市政府进行了一次专项系统**数据库调研**。调研中发现以下问题：

1. 其各下属机构的系统**比较分散**，有300余项数据库系统，其中包括A市资金（基金）综合管理系统、A市社会保险管理系统、A市人力资源与社会保障决策分析系统、A市劳动保障网上办事系统、A市退休人员管理系统等，“数据孤岛问题”严重。
2. 数据库管理系统的**类型复杂**，包括MySQL， Oracle， SQL SERVER， DM Database Server，HSQL DB，postgresql，MongoDB等。
3. 有大量市民反映其**信息登记系统繁多**，经常需要**重复填写个人信息**，且**有特定的需求就必须登录特定的系统**，他们希望有一个统一的平台能够办理全部业务，且该平台能在移动端使用。
4. 政府部门的数据库系统维护人员也不断反映，**与社保相关的系统过多，数据冗余现象严重**，数据库之间的数据交互也很难实现。
5. 政府负责人希望，能够有一个平台能够**自动生成相关数据报表与数据分析结果**供其查阅，最好能给出一些**智能化决策**建议。

A政府主管希望构建一个大数据应用系统，以系统全面解决上述问题，请你为上述目标系统提供一个架构设计方案。

要求

1. 请你结合上述背景，采用场景化方法，明确定义目标系统应该具有的质量属性，例如可获得性（availability）、安全性（security）、易用性（usability）与其他质量属性（X-Ability）。
2. 请你阐述为了实现这些质量属性，而采用的具体战术与架构模式。
3. 请你基于UML建模方法，给出Len Bass架构三视图，包括：模块分解视图、组件连接件视图和分配视图，可视化描述目标系统的架构设计方案。
4. 请你提供规范的架构设计规约，详细说明你的设计思路、设计目标、设计约束条件以及其它关键设计细节。
5. 以IBM数据科学虚拟实验室（datascientistworkbench.cn）为平台，以Jupyter Notebook为交互式开发环境，进行与命题相关场景的质量属性相关的实验性研究，以展示大数据分析的价值（可任意使用Python、R或者Scala语言）。

分析

这里逐题分析该怎么做。

1. 可以从如下方面出发
    1. 可获得性(availability)： 当用户(市民)更新数据时，数据库能及时相应数据更新请求。
    2. 互操作性(interoperability)：虽然我们需要合并大量数据库，但并不是合并得越精简越好，数据库之间要支持方便快速地查询其他数据。

3. 性能(performance)：当有许多用户在使用数据库时，要考虑到高并发的问题。
4. 安全性(security)：允许有权限的用户获取应该获得的数据，不允许数据被非法用户得到。同时还要保护数据传输，要足够安全。
5. 易用性(usability)：数据库接口应尽量避免冗余，命名和传输参数要规范，API要做到统一。
6. 其他质量属性你们看着吹，有道理就行。

2. 战术很多，架构模式也很多，这里只是举例子展示可以怎么写，你们要结合实际例子来稍微修改一下下面的语句。如果发现我写的目标跟你们的对不上，看看怎么修改(包括战术)能接起来。架构模式要画图，晚点再搞。

    1. 可获得性：

        1. 系统目标：数据库能及时响应数据更新请求。
        2. 实现战术：

            1. 服务后台时刻监控数据库状态，确定数据库是否可用及其网络状态是否畅通。
            2. 采用暖备份。平时只有保护组的活动成员处理数据更新请求。一旦活动成员发生故障，备用成员立即唤醒并继续工作。
            3. 绑定状态更新，保证在分布式组件之间交换的异步消息具有原子性和一致性。
        3. 架构模式：

            // 等会再说

    2. 互操作性：

        1. 系统目标：数据库之间要能方便快速地查询其他数据。
        2. 实现战术：

            1. 时刻定位其他数据库的网络位置，并保证网络连接畅通。
            2. 为每个数据库定制同一个中间层进行封装，并提供统一的接口设计。
            3. 使用SOAP协议来使得不同数据库之间传输的数据包尽量小。同时指定交互格式，通过统一数据交互协议进行数据库之间的交互。
        3. 架构模式：

            // 等会再说

    3. 性能：

        1. 系统目标：满足多用户使用数据库时，数据库仍然能保持高响应能力，且数据处理同步及时不丢失。
        2. 实现战术：

            1. 改进关键领域算法的时间复杂度，使得延迟降低。如数据库的查询等。
            2. 投入资金，配置更好的服务器，增加服务器建设资源。
            3. 尽量并行处理任务，减少额外阻塞的时间开销。如通过处理不同线程上的不同事件流来实现并发性。
            4. 允许一定的数据冗余，这样使得用户可以在不同的数据库查到所需的信息，加快查询速度。
        3. 架构模式：

            // 等会再说

    4. 安全性：

        1. 系统目标：保护系统数据免受未授权访问，同时提供被授权的人和系统的访问。
        2. 实现战术：

            1. 建立完善的防火墙并关闭危险的服务，将系统内的网络流量或服务请求模式与存储在数据库中的一组签名已知的恶意行为模式进行比较。
            2. 验证消息完整性。使用诸如校验和和哈希值之类的技术来验证信息、资源文件、配置文件的完整性。
            3. 检测消息延迟。检查传递消息所需要的时间来检测可疑行为，如中间人攻击。

4. 识别、认证并对参与者授权，确认远程用户或系统的身份并使得他们能有效访问数据和使用服务。
5. 对关键数据进行加密，如RSA加密算法等。
6. 及时锁定敏感资源，限制对敏感资源的访问。

3. 架构模式：

   // 等会再说

5. 易用性：

1. 系统目标：数据库接口应尽量避免冗余，命名和传输参数要规范，API要做到统一，使得数据库服务尽量简单易用。
2. 实现战术：
   1. 尽量将低级对象聚合到一个组内，减少重复操作。
   2. 维护任务模型，确定上下文，以便系统可以知道用户在做什么并提供帮助。
   3. 维护用户模型，明确表示用户对系统的了解程度以及用户在预期相应时间方面的行为。
   4. 维护系统模型，用于确定系统预期行为，以便向用户提供适当反馈。
3. 架构模式：

   // 等会再说