初步分析

c&c（Component and Connector） pattern汇总与分析

1. Broker，解决分布式计算服务的问题，使得客户和服务松耦合，提供服务器的透明性。任何分布式服务提供者与客户端都一定程度上使用了broker，soa依赖于broker。书上例子：ejb .net
2. ~~Mvc，主要目标~~：隔离界面的变化，使得界面修改维护不影响其他部分。书上例子：java swing asp.net ，可以不那么考虑。
3. ~~pipe-and-filter~~，针对离散数据流的处理，可以拜拜。书上例子：rss, workflow engine, 科学计算
4. client-server，不同的客户想访问共同的资源和服务，管理共同资源，分离共同服务，可以分布式分散资源。书上例子：web(http, browser), atm
5. ~~peer-to-peer~~，管理互相平等的分布式计算资源，为不同分布的用户提供服务。可以拜拜，例子：迅雷，utorrent, skype
6. soa，保证服务提供者的透明度（分布式／跨平台／跨语言／跨组织），是的服务提供者与使用者完全独立，可以是不同组织的不同系统。

Component：服务提供者，服务中间商，服务注册管理

Connector：soap, rest, async messaging。书上例子：adventure builder(用户使用这个系统完成度假规划／宾馆预订／付款等功能，本系统需要和很多外部系统合作提供完整的服务，使用http/soap通信)

强力推荐

1. ~~publish-subscribe~~，保障信息的消费者与提供者之间的完全透明性，而且不能保证信息的送达与可靠，可以拜拜
2. shared-data，多个不同计算组件需要操作相同的数据，data accessor访问数据，data store存储数据。可以考虑。

初步8选1 broker，4 c/s，6 soa，8 shared-data

根据质量属性再次考虑