MapGIS 10 技术架构白皮书

老子说“道生一、一生二、二生三、三生万物”。在业务愿景的技术实现过程中，假设“道”为愿景、一为方向、二为战略的话，三就应该是架构了。架构既出，万物化生可矣。战略是整体的、长期的。让架构直接承接战略，带来的最大好处是可以得到一个整体的可持续发展的系统平台。

本白皮将阐述 MapGIS 10 的技术架构设计，以及在设计过程中我们的思考。

1 “好”架构的定义

“好”系统的定义：能够满足业务需求的系统，它让业务能够随着时间的变化灵活地改变，并且用户会认为系统能够很好地支持业务。好的系统依托于优秀的架构，“好”架构应该满足以下特点：

1、架构必须能够达到功能与质量要求，以合理的研发、运行与管理成本产生使客户满意的系统，这是一个基本要素。但仅仅满足功能与质量要求的架构，还谈不上成功。

2、在长距离赛跑中，能够让系统以轻松的步伐始终跑在业务前面，是成功架构的一个显著特点。

3、成功架构的另一个显著特点是能够对公司的核心竞争力形成有力的支撑。

归结为一点，是静态架构转变成持续变化的架构的思想，持续业务转型的思想。

全球 80%以上的信息都与空间位置相关，空间信息也逐渐被政府、企业、大众所熟识、重视。空间信息项目的建设往往需要与众多业务领域的软件系统间建立联系和合作，而非单独实施、独立存在，并且也越来越趋向于基础建设方向。这就要求地理信息系统可以部署在更广泛的分布环境中，易于集成，易于重构，快速适应业务变化等。因此地理信息系统更加需要持续变化的架构思想，所设计的系统应该更具柔性。

悬浮倒挂式系统架构设计思路是在传统的业务层和技术层之间增加一个服务层，服务层通过一套协议或规范把功能从底层技术层调出来，加以封装，再根据业务层需求灵活组合。服务层不依附于任何特定技术平台，能够在业务层和技术层之间沟通、组合，业务应用系统

就变成了“松耦合结构”，想用什么功能就调用什么功能，需要什么功能就装配什么功能，改动调整非常方便。而且这些构建在各种各样系统中的“服务”可以以一种统一和通用方式进行交互。保证系统灵活性，另外，还可以保证“服务”的重复利用。

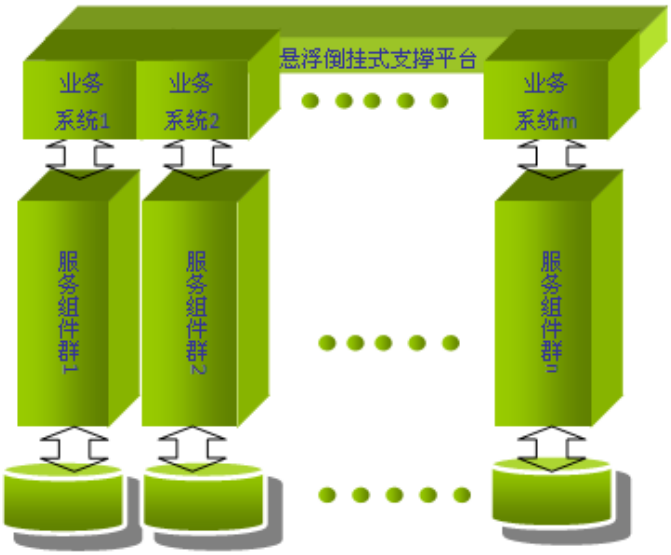
所以 MapGIS 10 平台采用了悬浮倒挂式系统架构设计的思想进行设计。

2 MapGIS 10 架构

2.1 悬浮倒挂式体系架构

2.1.1 悬浮倒挂式体系架构

MapGIS 10 的体系架构采用悬浮倒挂式的平台架构，是一种松耦合的面向服务的体系架构（如下图）。它与传统的奠基式向上支撑的平台架构有本质的区别；在这种体系架构下开发的系统牢固可靠，能很好的满足系统各功能高内聚、松耦合的要求，真正做到数据、功能和模型等全共享。



悬浮倒挂式支撑的平台架构

悬浮倒挂式体系架构是一种粗粒度、松耦合服务架构，服务之间通过简单、精确定义接口进行通讯，不涉及底层编程接口和通讯模型。悬浮倒挂式架构模式将应用程序的不同功能单元（称为服务）通过这些服务之间定义良好的接口和契约联系起来；这使得构建在各种这

样的系统中的服务可以以一种统一和通用的方式进行交互。在这种架构下，无数软件制造者可将它研制的软件功能以“服务”形式提供出来，各功能之间是相互独立的，以一种称为“松耦合”的协议机制来组合。因此基于悬浮倒挂式架构设计的系统易于扩展，能够适应不断变化的客户与市场需求，使开发者可将更多的精力转移到专业服务提供上。

悬浮倒挂式的架构思想传承于 SOA 体系的扩展，我们也可以称之为悬浮式的 SOA 体系架构，悬浮式 SOA 体系架构有两个鲜明的特点：

第一个，可伸缩性。资源可以聚合，根据不同需要、根据不同业务属性进行聚合，可以重新定制应用。

第二个，自适应性，规模可以动态伸缩，可以满足应用大规模增长的需要。细胞可以随时替换，一个云细胞出了问题，新的云细胞可以随时聚集过来，提供无限多的千变万化的应用。

悬浮式 SOA 体系架构性能是一种柔性架构，是动态平衡，理论上在这种架构下可以无限扩展。

采用基于悬浮倒挂式的系统架构构建 MapGIS 10，从框架层上保证系统集成的可行性、系统的易扩展性，以及灵活性。基于悬浮倒挂式的 MapGIS 10 架构的优势有如下：

(1) 开发灵活性

MapGIS 10 可基于模块化的底层服务、采用不同组合方式创建高层服务，从而实现重用，这些都体现了开发的灵活性。此外，由于服务使用者不直接访问服务提供者，这种服务实现方式本身也可以灵活使用。

(2) 更易于集成和管理

在面向服务的体系架构中，集成点是规范而不是实现；这提供了实现的透明性，并将因为基础设施和实现发生的改变带来的影响降到最低限度。通过提供针对基于完全不同的系统构建的服务规范，使应用集成变得更加易于管理。

(3) 更易维护

服务提供者和服务使用者的松散耦合关系及对开放标准的采用确保 MapGIS 10 对该特性的实现。

(4) 更好的伸缩性

依靠服务设计、开发和部署所采用的架构模型实现伸缩性。服务提供者可以彼此独立调整，以满足服务需求。

(5) 降低风险

利用现有的组件和服务，可以缩短软件开发生命周期（包括收集需求、进行设计、开发和测试）。重用现有的组件降低在创建新的业务服务的过程中带来的风险，同时也可以减少维护和管理支持服务的基础架构的负担。

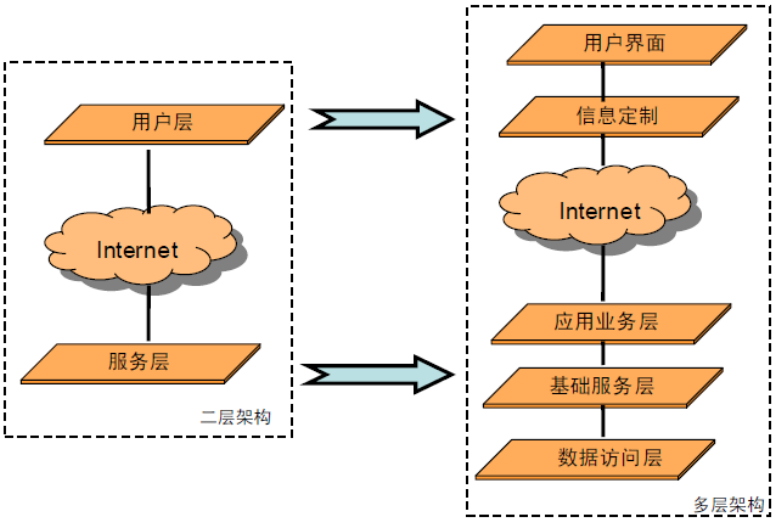
(6) 更高的可用性

该特性在服务提供者和服务使用者的松散耦合关系上得以体现。使用者无须了解提供者的实现细节，这样服务提供者就可以在多种集群环境中灵活部署，使用者可以被转接到可用的例程上。

2.1.2 多层体系架构

基于悬浮倒挂式架构思想，MapGIS 10 构建多层体系架构。

多层架构的核心思想是，将整个业务应用划分为表示层－业务层－数据访问层－数据库，明确地将客户端的表示层、业务逻辑访问、和数据访问及数据库访问划分出来，十分有利于系统的开发，维护、部署和扩展；同时业务层又可以根据系统实际需要再度细分为平台服务层和应用业务层等。两层和多层的体系结构比较如图：



多层体系结构示意图

使用多层体系结构设计可以达到如下目的：分散关注、松散耦合、逻辑复用、标准定义。好的多层次结构，可以使得开发人员的分工更加明确；每个层次由专门的程序员负责，各个程序员只用关注本层次的实现方式；一旦定义好各层次之间的接口，负责不同逻辑设计的开发人员就可以分散关注，齐头并进。例如 UI 人员只需考虑用户界面的体验与操作，而应用业务层的开发人员仅关注业务逻辑的设计与实现，而数据访问层的设计人员也只用处理数据库各个业务表的访问，不必关注业务数据的流转。建立在多层次架构上的系统，具有很好的扩展性，当有新的需求变动和需求增加时，只需要修改应用业务层或用户层，而不必涉及到系统底层的实现。

松散耦合的好处是显而易见的。如果一个系统没有分层，那么各自的逻辑都紧紧纠缠在一起，彼此间相互依赖，谁都是不可替换的。一旦发生改变，则牵一发而动全身，对项目的影响极为严重。降低层与层间的依赖性，既可以良好地保证未来的可扩展，在复用性上也是优势明显。每个功能模块一旦定义好统一的接口，就可以被各个模块所调用，而不用为相同的功能进行重复地开发。

多层体系结构可以增加系统开发的效率，同时降低服务器的负载，但是由于层次结构明确，上层的业务访问必须经过其下层的功能支持，因此要实现多层体系结构必须解决几个问题。

1、进行好的分层式结构设计，标准也是必不可少的。只有在一定程度的标准化基础上，这个系统才是可扩展的，可替换的。而层与层之间的通信也必然保证了接口的标准化。层间

接口设计的标准化需要解决。

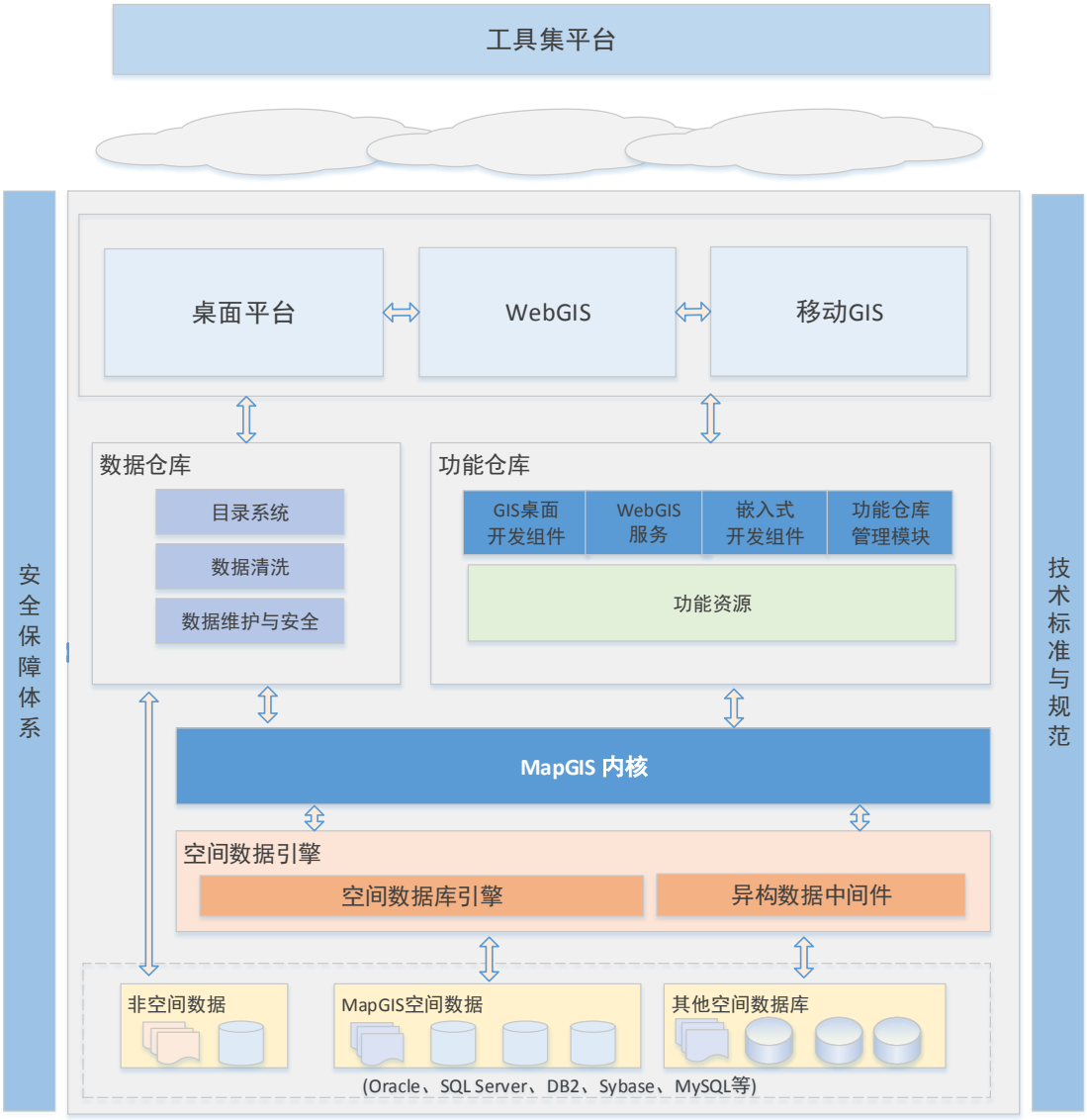
2、如果不采用分层式结构，很多业务可以直接造访数据库，以此获取相应的数据；采用多层体系结构后却必须通过中间层来完成，怎样提高这种间接访问数据库方式的效率非常重要。

针对以上的问题，我们在设计多层体系结构的同时，设计严密的接口规范。对接口的定义、参数的说明、代码习惯、变量命名规则、接口功能描述等内容都有详细的记录，以保证标准的实时和有效。而对于多层结构所带来的数据库访问效率问题，我们在设计时将数据库访问层接口的性能设计放在第一位，，同时通过合理优化数据库来保证数据库访问的高效。

2.2 MapGIS 10 架构图

以上内容已经对 MapGIS 10 所采用的面向服务架构、多层体系架构进行分析，其优势明显，构建目标明确。本节在该基础上，对 MapGIS 10 的详细结构进行说明。

MapGIS 10 由空间数据引擎模块、MapGIS 内核、数据仓库模块、功能仓库模块、桌面平台、WEBGIS 平台、移动 GIS 平台构建，支持多种方式的桌面 GIS 开发、WebGIS 可视化发布应用和移动设备的应用。MapGIS 10 体系结构如下图所示：



MapGIS 10 总体结构示意图

如上图所示，我们在传统的架构上加入更为细化的多层架构模式。引入数据仓库、功能仓库层封装底层的功能支持，功能是以服务的方式提供，功能的接口支持标准协议，实现 MapGIS 10 跨平台、跨系统的高度灵活性与应用性。而负责业务开发的人员可以在底层的基础上做更为灵活的模块开发和搭建，从而增加工作效率以及系统的扩展性。

在实际的应用中，随着开发项目领域的扩展，功能库是不断被丰富的，并在功能仓库中被统一管理、维护；数据仓库负责访问存放于各分布的服务器、工作站、主机上的数据资源；基于构建和运行环境，用户利用功能仓库、数据仓库提供的资源，进行搭建、配置式二次开发，得到具体业务应用系统的解决方案并运行，最终提供模型和异构数据表现和信息可视化功能，运行于多种操作系统上。

MapGIS 10 采用这种架构设计可实现：支持分布式数据存储，提供集成化开发；提供统一数据管理平台，支持子系统相对独立运行；开发的应用系统适用稳定，能够充分满足业务需求；采用基于 GUID 资源转换和元数据过滤规则形成安全的数据库和安全的功能库的模式，保障数据的安全性；提供当前最新的搭建式、配置式、插件式二次开发技术，以最快的方式构件应用系统。

在创建大型信息化解决方案时，一个解决方案通常包括多个业务领域的应用，产品功能和结构都非常复杂。MapGIS 10 同时支持 C/S 架构和 B/S 架构，支持部署在多种硬件设备上，能够极大地增加软件系统部署和运行的灵活性，并降低软件系统的开发和维护成本。

此外，我们还推出了工具集平台，类似于移动端的 Appstore，让所有的 GIS 开发者，利用我们提供的 SDK、Web API、mobile API，以及各种便捷的开发方式，去开发自己的 GIS 应用。开发者只需要注册、上传、出售发即可完成，而使用者可以直接在上面搜索、购买、下载使用。

2.3 MapGIS 10 架构模块详解

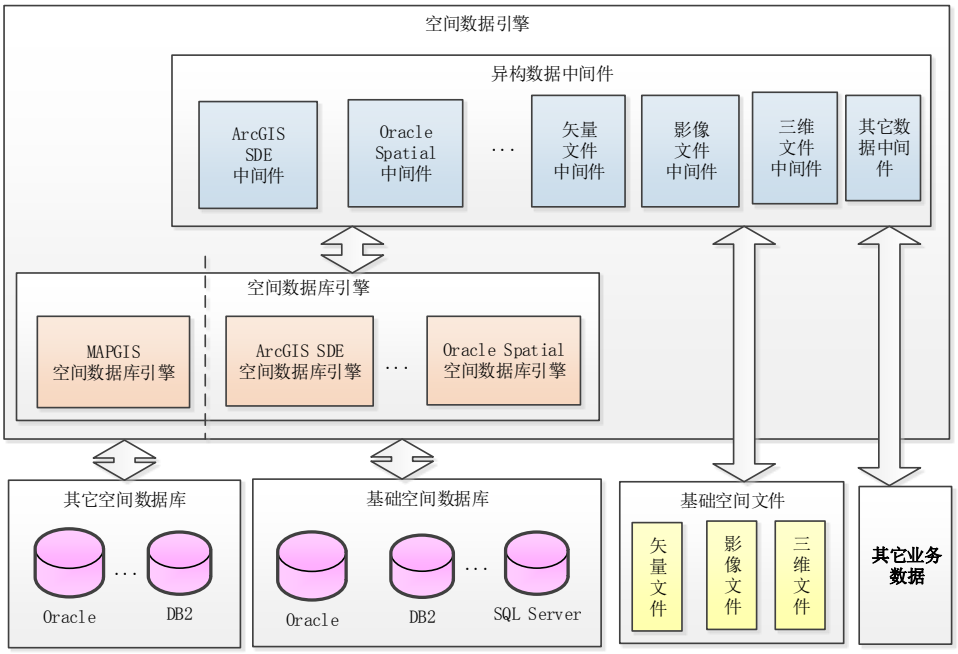
MapGIS 10 大模块之间相互独立，以“松耦合”的协议机制来组合。

2.3.1 空间数据引擎

空间数据引擎模块是 MapGIS 10 与其它系统通信的数据接口。空间数据引擎采用中间件的形式将多源异构的矢量文件、影像文件和其他业务数据映射为统一的面向实体的空间数据模型；并通过空间数据库引擎按照面向实体的空间数据模型提供对矢量空间数据、影像空间数据和其它业务空间数据的统一数据库存储；设计具有高度的抽象的虚拟空间数据引擎，对外提供统一的空间数据引擎接口，实现数据直接读写。

1. 空间数据引擎总体结构设计

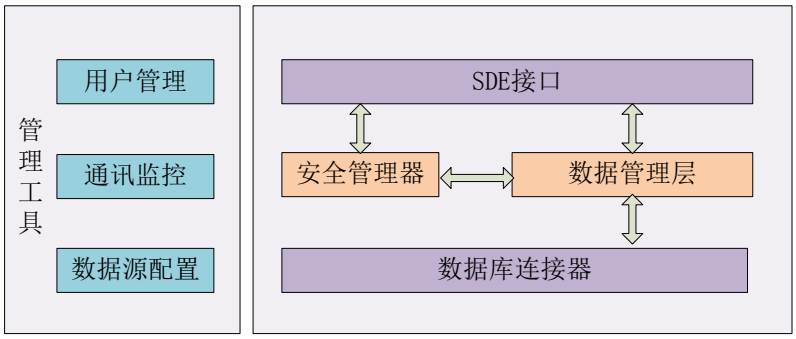
空间数据引擎包含如下子模块：空间数据库引擎、异构数据中间件。空间数据库引擎负责与各类商业及开源数据库通信；异构数据中间件负责各种类型的异构数据的转换，克服异构和分布带来的数据使用障碍，既可以保留数据异构和分布性的优势，同时也可以为更多资源共享、处理协同与任务合作方面的用户提供一致化的服务接口和方式。



空间数据引擎总体结构示意图

2. 空间数据库引擎

在 MapGIS 10 的空间数据引擎的体系框架中，空间数据库引擎按照面向实体的空间数据模型提供对矢量空间数据、影像空间数据的统一数据库存储，并建立适应海量数据存储管理的空间数据组织机制和空间索引机制。空间数据库引擎支持的商用及开源数据库包括 Oracle、SQL Server、DB2、等商用数据库以及 MySQL 等开源数据库。空间数据库引擎结构如图所示。

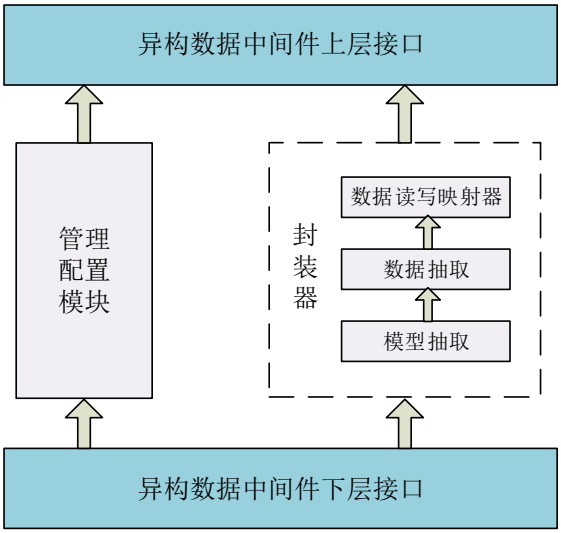


空间数据库引擎结构

3. 异构数据中间件

在 MapGIS 10 的空间数据引擎的体系框架中，通过异构数据中间件的形式实现对各种文件格式的异构数据直接访问能力。包括对 Arc/Info Geodatabase、Coverage、ShapeFile、

Mif、AutoCAD、E00、VCT 等矢量文件的读取；包括对 GeoTIFF、IMG、ENVI、PIX、MrSID、RAW、JPEG、JPEG2000、HDF、NetCDF 等影像文件的读取；包括对 AutoCAD、3D MAX 等三维文件的读取等。

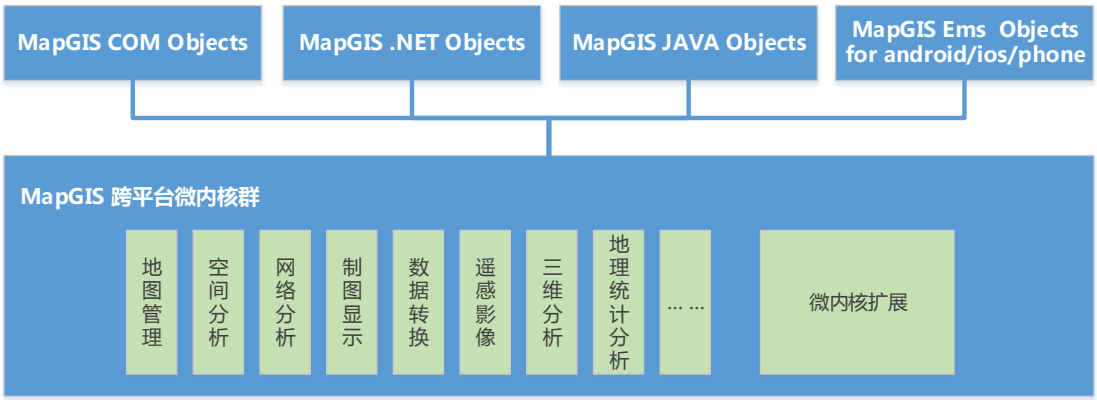


矢量数据中间件结构

2.3.2 MapGIS 内核

MapGIS 内核提供跨平台数据存储与管理及空间分析、可视化等 GIS 功能；提供操作系统屏蔽层，支持在多种主流服务器操作系统上高性能运行；提供配置池进行全局配置，实现多进程多任务并发数据处理。

MapGIS 内核由跨平台微内核群与组件群构建。



微内核群提供数据存储与管理、空间分析、数据渲染、出图等 GIS 核心功能。微内核

群是 MapGIS 平台的基础，由标准 C++构建，是平台的核心技术。我们将传统软件内核中的许多部分移出内核，只留下最核心的功能组成微内核群，并采取服务器方式实现，向外界提供服务；微内核群层次分明，将黑性能算法按功能划分并分类封装，包括：地图管理、空间分析、网络分析、制图显示、数据转换、遥感影像处理、三维分析、地理统计分析等功能模块。模块之间，逻辑界限清晰，依赖关系简明，低耦合、高内聚。

在新一代的 MapGIS 10 平台中，提供了标准化程度更高、功能更完备、接口更规范的微内核群，这必将进一步增强 MapGIS 平台的跨平台、高效、及能够与硬件架构同步的能力。

组件群是在微内核群基础上进行的再次封装，提供用于支持快速应用开发的一套标准开发接口，可跨语言跨平台调用，可以很好的定制出桌面应用系统、网络系统以及移动端的应用。

2.3.3 数据仓库

数据仓库模块主要包含：数据目录管理子模块、数据清洗子模块、数据维护及安全子模块等。

数据仓库模块在目录系统上实现对数据的仓库式管理，能够以统一的方式集成管理二、三维空间数据、栅格数据、多媒体数据、文字标注数据、文档信息、数据库数据、元数据信息及各类业务数据，满足平台对多源异构数据的管理需求。

1. 数据目录管理子模块

数据仓库在对数据进行管理时，通过中间件机制规避数据本身的类型信息，实现多源异构数据的一体化目录式管理，并以单节点定制和驱动定制目录树两种形式展现，并可形成具备分层特点、充分满足业务需求的主题数据树。对于领域中自定义的数据类型，可以根据 MapGIS 10 提供的二次开发方式，以向导式开发的模式开发驱动插件集成到 MapGIS 10 的功能仓库中，实现对特殊数据类型的支持。

2. 数据清洗子模块

数据清洗子模块从来自多种数据源（异构的 GIS 数据、文档数据、影像数据）的数据库中将数据抽取、清洗、再入库。数据清洗不仅是简单的用优质数据更新记录，还应涉及数

据的分解与重组。

在数据清洗环节上，不单提供通用的 GIS 数据清洗（包括脏数据修改、数据检查、乱码清洗等），将脏数据转化为满足数据质量要求的数据；同时，提供可扩展的清洗机制，提供支持特定业务数据的自定义清洗规则接口，如插件的形式、基于工作流的流程。数据的清洗流程采用“分层”的设计思想：前面的处理层为后面的处理层提供相对“干净”的数据，后面的处理层基于前面的做进一步的清洗。

3. 数据维护及安全子模块

数据维护及安全子模块通过提供元数据定义、数据库索引表等维护管理功能来实现。

数据维护包括数据库维护、用户权限维护、工作流维护、统计图表维护等功能。数据安全包括数据权限管理和数据操作日志设计。通过数据权限管理验证机制，加强登录身份认证，确保用户对数据使用合法性，严格限制登录者的操作权限，保证用户数据的安全性。在数据操作日志服务和管理设计上，MapGIS 10 拟对所有客户对数据的操作进行记录，所记录的内容进行保存并不得删除。

此外，MapGIS 10 的权限管理体系具备良好的扩展性，不同的用户可以根据自己的数据安全需要开发设计相应的安全措施。

2.3.4 功能仓库

MapGIS 10 功能仓库实现功能资源在 MapGIS 10 的统一管理和直接以搭建的方式调用。这些功能资源包括基础功能资源、桌面开发组件、web 网络服务、嵌入式开发组件。另一方面，功能仓库实现功能资源的自定义视图管理：对于用户按照 MapGIS 10 提供的标准开发的针对特定业务领域的功能，既可以原有的功能资源一起形成二次开发用户自己特有的功能仓库；又可以从父功能仓库中提取用户真正关心的功能资源，按照领域组织子功能仓库。

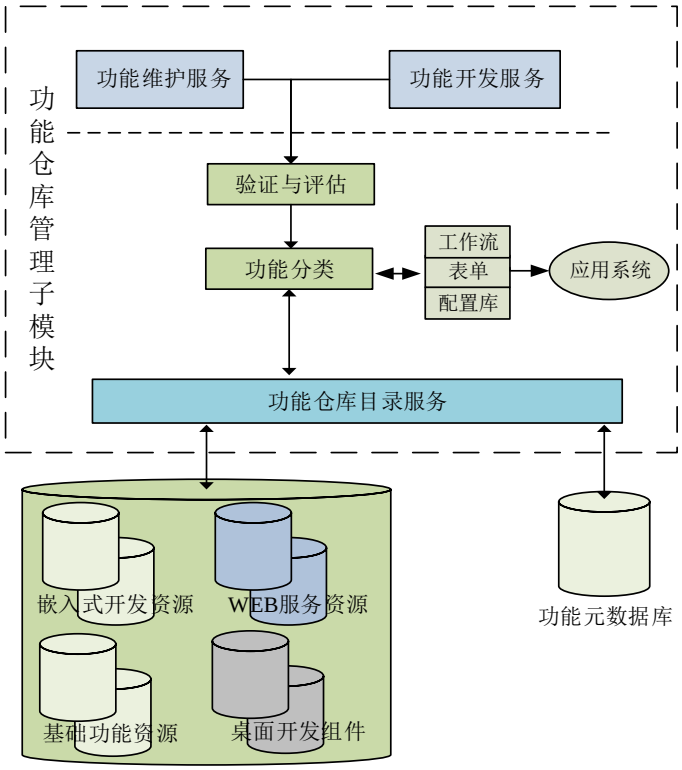
MapGIS 10 功能仓库功能资源支持“热插拔”搭建。无论是功能插件资源、组件资源，还是基于两者搭建的流程资源，都可以在 MapGIS 10 框架下实现直接调用；MapGIS 10 功能不依赖于某一种开发语言，可以直接嵌入到通用开发环境中实现 GIS 功能，而其他的专业模型功能也可以使用这些通用开发环境来实现，也可以插入其它的专业性模型的分析控件，各个模块之间可以实现高效、无缝的系统集成。

功能仓库由三个子模块构建：功能仓库管理子模块、功能资源库、功能元数据库。

功能仓库通过制定标准的协议统一管理来源于构件库的异构功能资源，并依托于目录树的层次性对这些功能资源进行有效的分类查询、检索、管理；检索出的功能项通过工作流灵活定制功能粒度，通过表单实现相应 Web 发布界面，提供配置库实现系统非界面元素的配置，最终达到通过搭建、配置的方式开发应用系统的目的。

功能维护服务和功能开发服务增加针对特定业务领域的功能仓库的扩展性。

功能元数据仓库提供方便的用户工具，如服务方法管理、功能检索、用户查询。这些工具以友好的界面形式与最终的用户交互，从而实现用户对整个功能仓库的可视化管理。



功能仓库总体结构图

2.3.5 桌面平台

桌面平台基于.NET 的插件式框架，以插件的形式进行组织，实现可扩展、易定制的 GIS 应用框架。我们将 GIS 功能进行分类组合，提供了数据管理插件、工作空间插件、地图编辑插件、栅格编辑插件、影像分析插件等 20 多个不同功能、不同粒度的插件，用户可以按

需选择，动态加载应用。

此外，插件式框架也提供了灵活的方式扩展方式，以支持不断变化的业务需求。

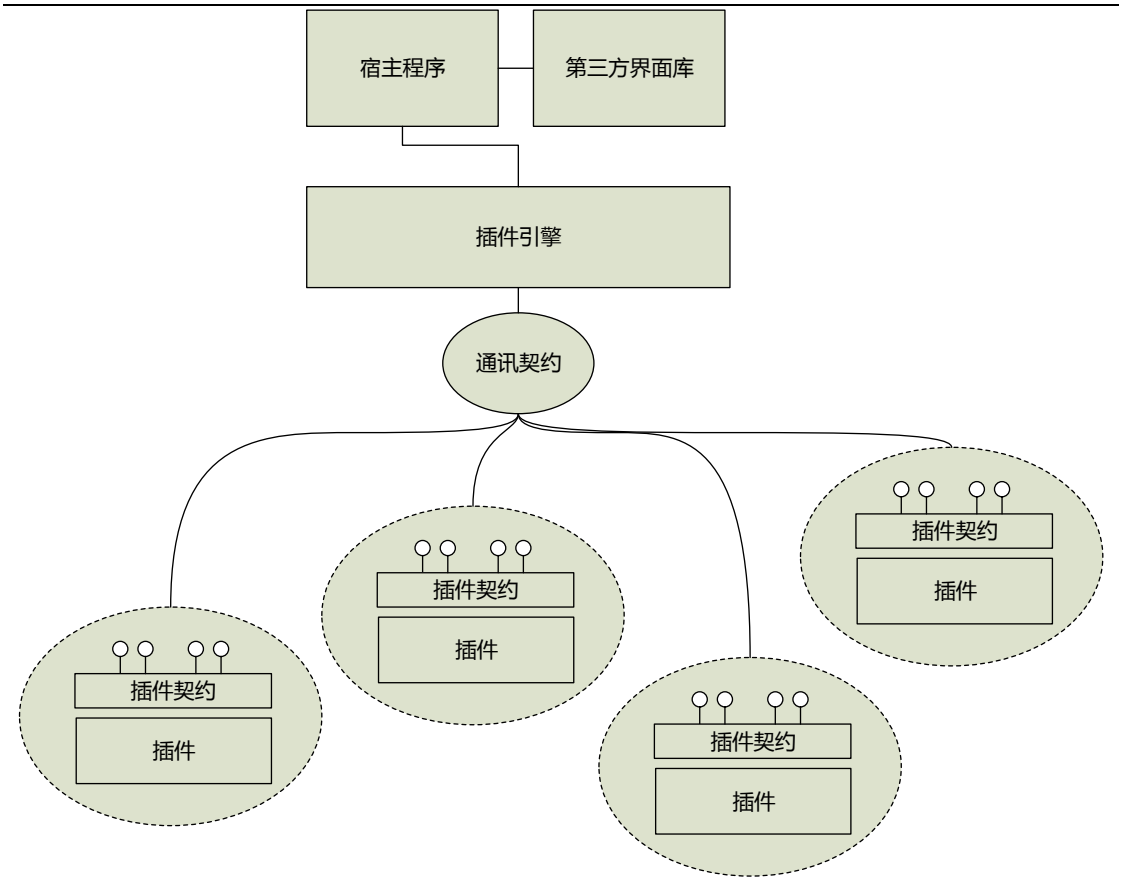
插件式框架主要由：插件引擎、宿主程序、插件程序集、第三方界面库组成。

插件引擎，负责解析插件程序集，提取程序集中的插件类型信息，并提交给宿主程序生成对应的界面对象。插件引擎提供一种通讯契约，即标准插件接口。插件程序集只要实现了这些接口，就能被插件引擎认可为插件。插件引擎提供一个插件容器(PluginContainer)，负责管理插件的加载，卸载等状态控制。插件引擎提供一个运行框架(Application)，负责管理插件引擎运行状态和与插件程序集间的交互。

宿主程序，是框架运行的入口，它通过插件引擎加载插件对象，并将插件对象 UI 形式来展示，并负责协调这些插件对象与界面控件间的交互。

插件程序集，是实现了插件引擎定义的插件契约的应用程序集，是基于.NET 框架的应用功能的主体实现。

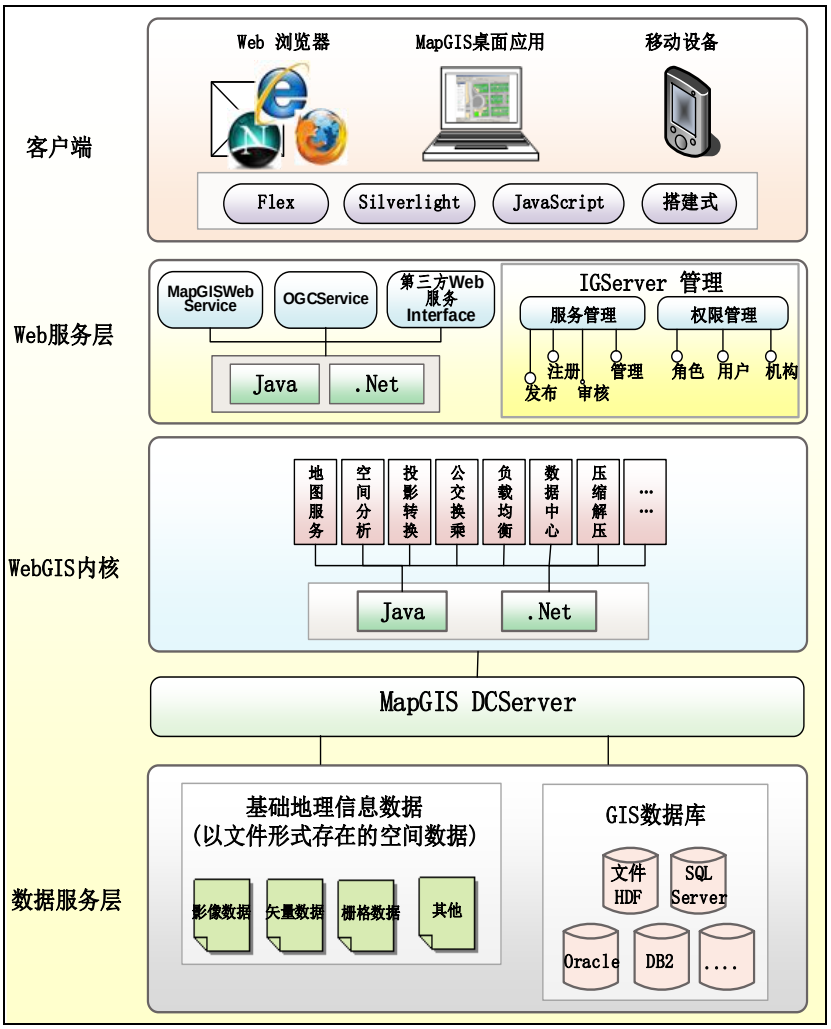
第三方界面库提供界面控件对象，被宿主程序调用来实现与插件对象间的交互。



2.3.6 WEBGIS

WebGIS 平台是一个面向服务的分布式 WebGIS 开发平台，提供跨平台的网络 GIS 服务和开发框架。

WebGIS 平台体系架构如下图所示：



WebGIS 平台体系架构

客户端：支持多种 Web 浏览器（如 IE、Firefox 等），支持各种 Web 应用程序的访问或嵌入到已有 Web 应用程序中。在客户端层面上，可支持四种开发方式，包括 Flex、Silverlight、JavaScript 和搭建式开发方式。用户通过客户端与 Web 服务层进行交互。

服务层：运行于 Windows/Linux/UNIX 等操作系统上，主要提供各种 Web 服务，包括 MapGIS WebService、OGC WebService 和第三方 Web 服务接口，其中 MapGIS WebService 和 OGC WebService 分别提供.NET 和 JAVA 两个不同的版本。客户端通过浏览器或者其他的方式（桌面应用等）向 Web 服务发送请求，Web 应用服务进行响应并接收请求，返回相应的操作结果。

WebGIS 内核：主要负责与数据服务层的数据通信，主要提供.NET 和 JAVA 两种版本的内核。客户端发送数据请求，通过内核实现与数据服务层的通信，将数据返回到客户端缓存。

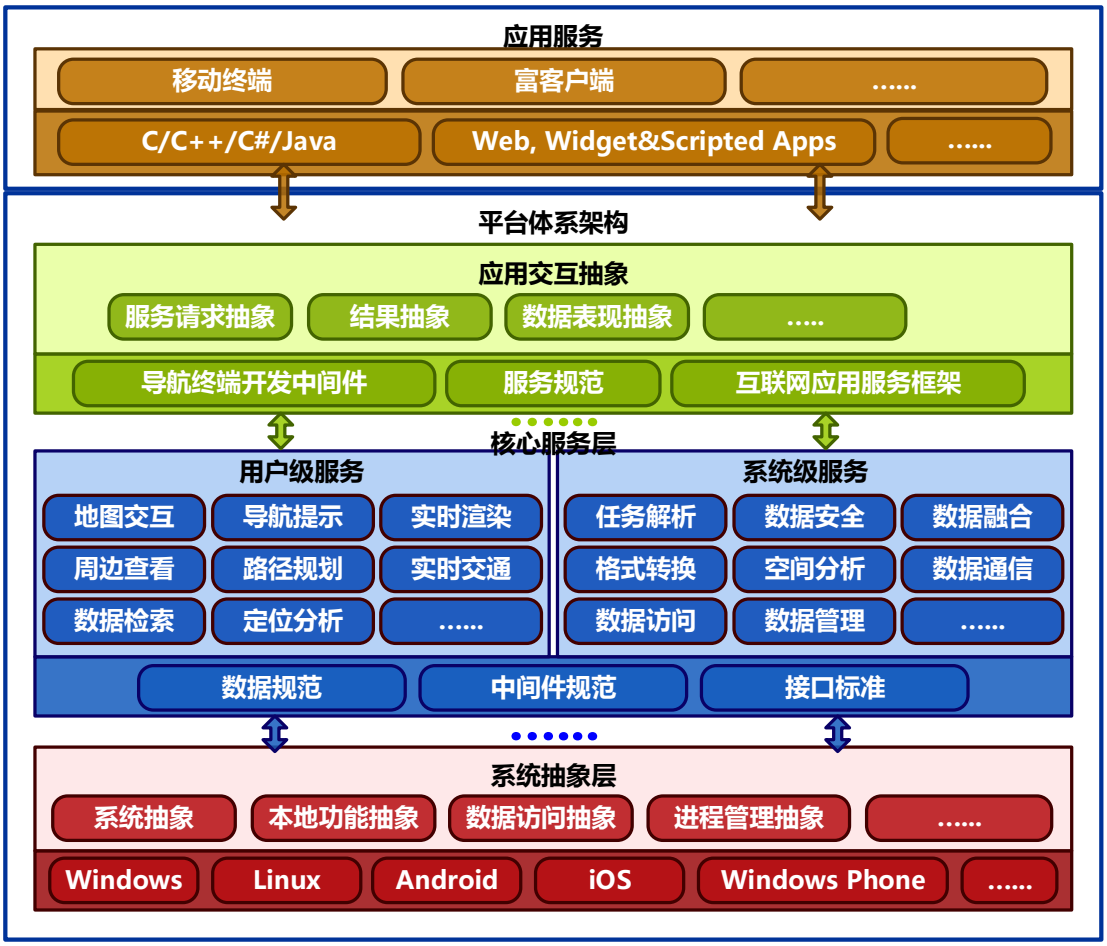
基础地理信息数据和数据库中存储的数据可以通过 GIS 服务器通信处理数据请求，将处理后的结果返回给客户端。

数据服务层：包括 GIS 数据库中的数据和基础地理信息数据。GIS 数据库中包含以统一的 MapGIS 数据格式（HDF 方式进行存储的 GIS 数据）的数据以及其他数据库存储的数据（例如 DB2、Oracle 等）。基础信息地理数据包括：影像数据、矢量数据、瓦片数据等，它们都是以文件形式存放的空间数据。web 的数据调用都是基于平台，这充分发挥了平台管理海量数据能力和并发访问数据能力。

2.3.7 移动 GIS

移动 GIS 平台，依托 WEBGIS 平台在服务器端提供丰富的地理信息空间信息服务支持，实现移动端的信息服务共享，面向行业和大众领域，提供无差异的在线和离线式 GIS 服务，构建完整的行业解决方案。

移动 GIS 平台的架构如下图所示。



中间件平台体系架构

在中间件平台体系架构中，最上层是应用交互抽象层，在这一层中，用户的各种服务将会按照规范所定义的接口形式发送到中间件体系中，首先处理这些请求的就是应用交互抽象层。该层根据用户的请求将请求进行分类，根据服务所属的类别，并最终提交给核心服务层。

核心服务层是由用户级服务和系统级服务组成，它们分别承担了两种类型的服务，第一种是针对导航服务中的用户级服务，这些服务以具体的用户服务需求为最大特征，如周边查看，路径规划，实时交通，定位分析等；第二种是以数据处理和空间分析为主要特征的服务，这些服务没有非常明确的应用性，而是以功能性为主，如服务请求的分解，数据的安全处理，数据的综合，格式的处理等。

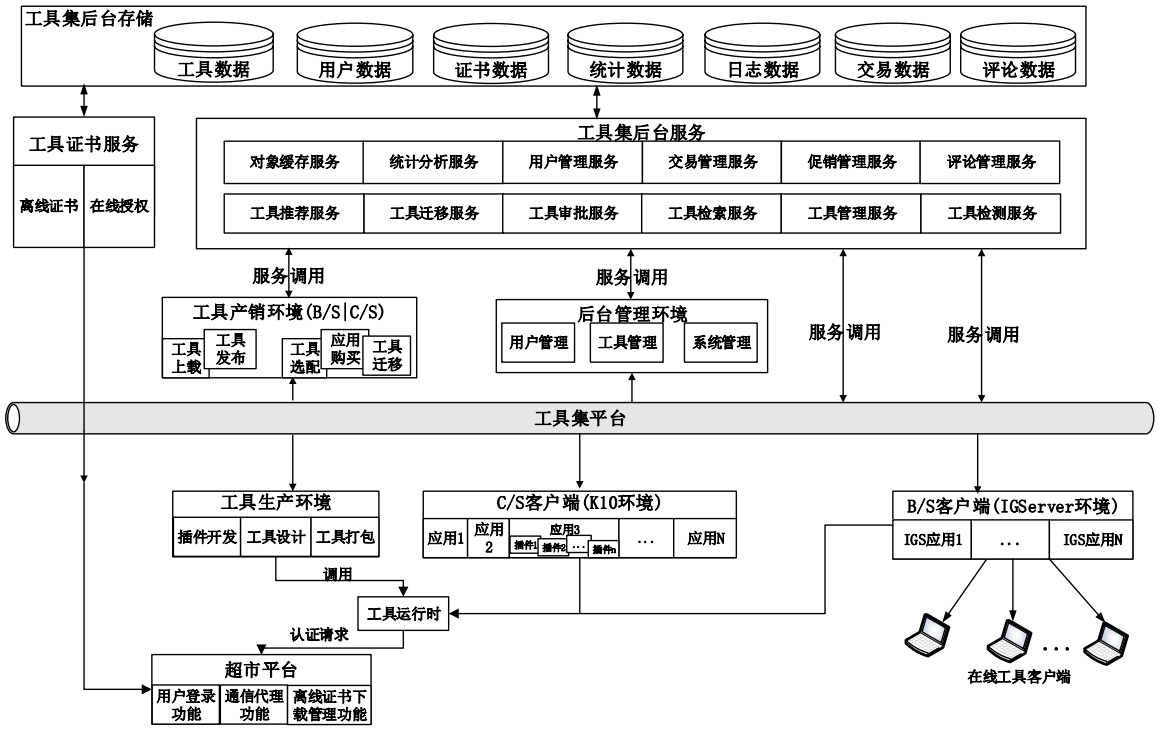
最下层是系统抽象层，这层的功能是为了实现具体的服务的功能的，如在导航终端中的运行时的地图渲染最终要反馈到移动设备上进行执行，在富客户端中也是需要最终调用相应的系统级接口来实现，这些实现必须与是实际的应用特征相关联，即需要一个抽象层来与这些硬件设备来进行交互，通过统一的接口规范，针对不同的设备，进行不同方式的处理和解

决，使硬件的差异对用户完全透明。

2.3.8 工具集平台

工具集平台，类似于移动端的 Appstore，让所有的 GIS 开发者，利用我们提供的 SDK、Web API、mobile API，以及各种便捷的开发方式，去开发自己的 GIS 应用。开发者只需要注册、上传、出售发即可完成，而使用者可以直接在上面搜索、购买、下载使用。

工具集平台主要包括超市平台、工具生产端、应用运行端、后台服务端、后台管理端、证书服务端等主要功能模块，最终构建云环境中基于在线/离线等认证方式控制下的工具聚合(生产)、迁移(上载/下载)、重构（安装）等功能的 GIS 工具程序产-销的新型生态环境。

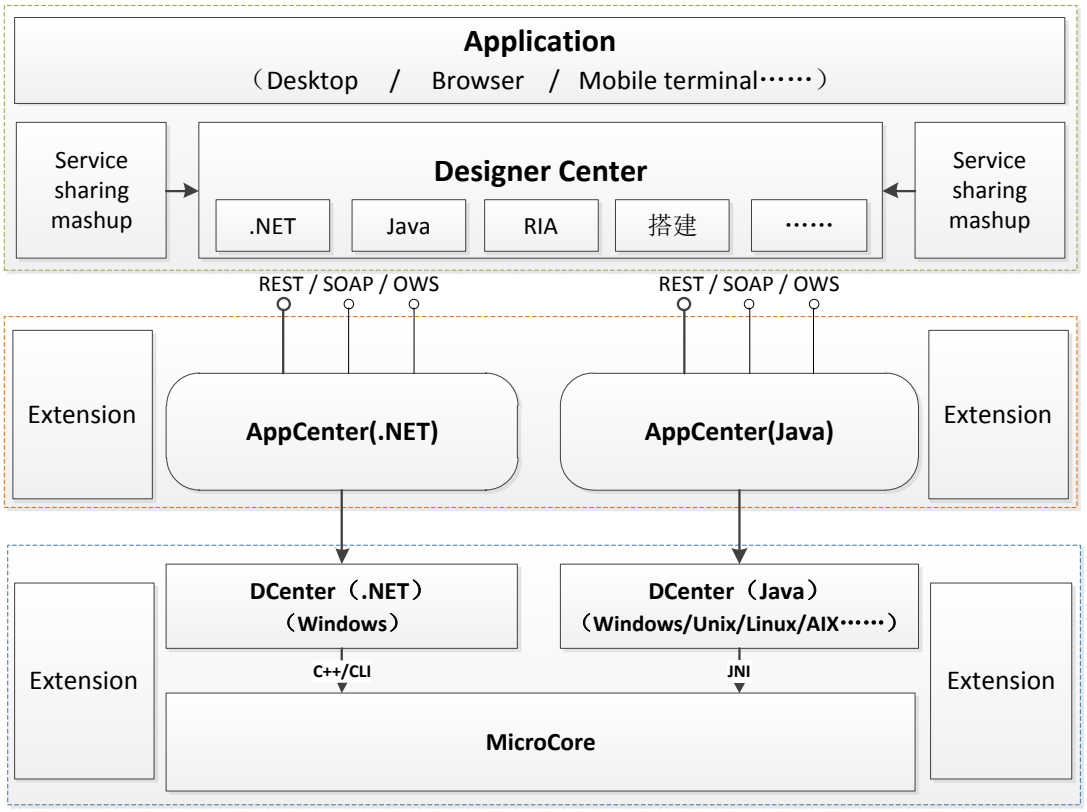


3 MapGIS 10 二次开发体系

MapGIS 10 根据 java 与.net 两条技术路线，基于 MapGIS 内核，提供统一的二次开发框架与扩展机制，满足各行业领域的各种层次的应用需求。

在基础内核层、GIS 服务层、应用层，每层均提供扩展机制，全面支持桌面应用、WEB

浏览器应用和移动设备等应用。



从技术角度观察，支持两种二次开发模式，即 C/S 开发、B/S 开发。

C/S 开发：即客户机与服务器结构的开发方式。基于平台提供的二次开发框架构建 C/S 模式的 GIS 桌面应用系统，客户须安装配置相应的 GIS 平台与应用系统环境。此种开发模式可以充分发挥硬件环境的优势，应用系统具有较高的性能，但无法满足互联网环境的共享访问使用。

B/S 开发：即浏览器和服务器结构的开发方式。平台提供多种 B/S 开发框架，即主流 RIA 的纯客户端开发、JAVA 与.NET 体系的开发等。基于平台二次开发框架开发 B/S 的 GIS 应用系统，通过 WEB 服务器部署，网络内的任一客户机通过浏览器即可进行访问，没有软件环境的限制。B/S 开发的系统维护简单，成本低，可以轻松实现网络内的资源共享，且使用简便，目前已成为主流的 GIS 应用模式。

3.1 C/S 开发框架

C/S 的开发框架提供多种开发的接口，可以适应.NET，Java 和 C++等开发环境。开发者可以使用这些组件来开发和 GIS 相关的地图应用，包括从简单的地图浏览到高级的 GIS 编

辑程序。

目前，C/S 程序主要包括单机应用与网络应用，随着互联网技术的不断发展，基于 C/S 的网络应用已经逐步取代单机应用，如网络游戏、QQ 等，都是典型的 C/S 网络应用程序。互联网也慢慢向各行业领域渗透，GIS 行业也不例外。GIS 行业经过多年积累，已经积累了大量的功能、数据、客户资源信息，如何将这些信息更好的利用到 GIS 产品中，打造一个大型的共享平台，以及如何高效率、高性能进行部署与使用，也是 GIS 平台当下需要解决的问题。

MapGIS 10 所有功能都是以服务的形式提供的。基于 MapGIS 10 可轻松开发基于网络应用的 C/S、B/S 程序，并且提供基于.NET 与 JAVA 两大技术体系的研发 API，以满足不同用户层的需求。C/S 开发框架设计模式如下图所示：

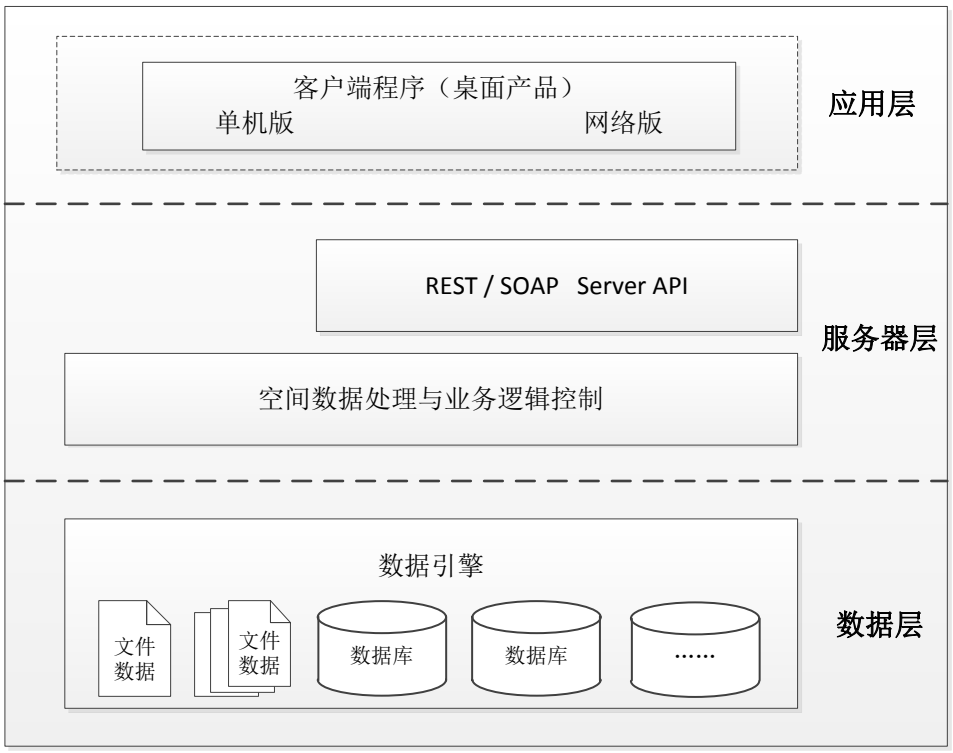


图 开发设计模型

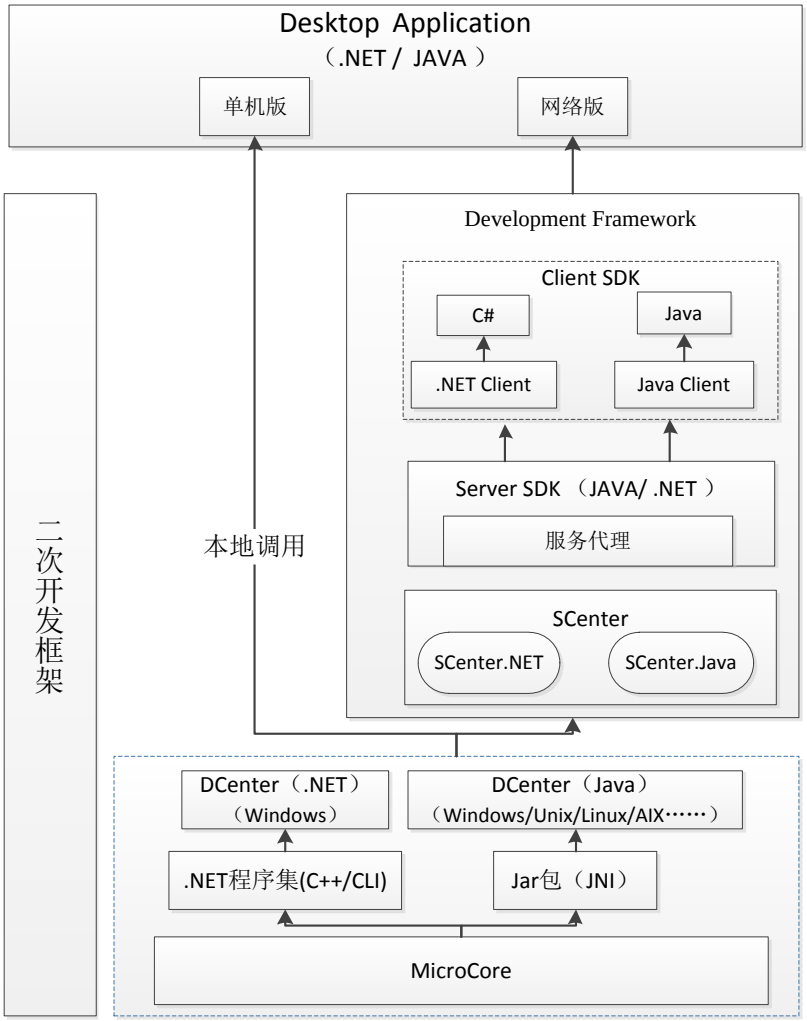
数据层：支持文档类数据、GIS 数据等数据格式；提供文件驱动和中间件技术，来实现异构数据的统一管理；通过文件驱动技术，可实现 GIS 行业业务文档的管理，如 Word、PPT、PDF 等格式。通过中间件技术，可将异构 GIS 数据进行统一的管理，包括 ArcGIS、MapInfo、AutoCAD 等数据的管理。实现数据的仓库式管理。

GIS 服务器层，开发框架核心，采用微内核技术构建，基于微内核技术实现空间数据的处理与业务逻辑控制功能，可直接基于该模模块功能，开发单机版桌面产品。支持网络版桌面产品时，需调用 Web 服务来实现，因此，空间数据的处理与业务逻辑控制层的基本上再加上了“REST/SOAP Server API”层，提供 REST 和 SOAP 两类服务接口，借开发网络版桌面产品调用。

应用层：基于 GIS 服务器层提供的.NET 和 JAVA 的 API 接口，既可开发出对应的桌面产品。

基于上述设计思想、原则与设计模式，构建统一的 C/S 开发框架，为用户提供灵活、高效的二次开发解决方案。

C/S 框架开发包括 DCenter 层、SCenter 层、Development Framework 层、客户端层。DCenter 层提供平台的微内核，以及在微内核的基础上封装成的.NET 程序集和 Jar 包。Development Framework 层为开发框架层，针对 C/S 开发，提供了两类开发方式和对应的开发库，一类为基于.NET 框架的.NET 开发，主流开发平台为 Microsoft Visual Studio，主流开发语言为 C#.NET。另一类为基于 J2EE 框架的 Java 开发，主流开发平台为 MyEclipse，开发语言为 Java 语言。客户端层，为基于开发框架开发出来的桌面产品，这些桌面产品的功能都是以服务的形式提供，可面向网络应用。



C/S 框架开发框架

在开发.NET 网络版桌面程序时，基于.NET 开发平台（如 Microsoft Visual Studio）开发 GIS 功能时，调用通过服务代理处理后的 Server SDK for .NET 接口，采用的开发语言为 C#.NET 语言来完成.NET 网络版桌面程序的开发。也可直接在 VS 工程中调用服务地址（REST 或 SOAP）来完成功能开发。开发后的程序可运行在 Windows 平台之上。

同样，在开发 Java 桌面程序时，基于 Java 语言的开发平台（如 MyEclipse），采用直接调用 Server SDK for .Java 接口，或者直接引用服务地址的方式，采用 Java 语言来完成 Java 网络版桌面程序的开发。Java 语言具有跨平台的特性，开发后的 Java 网络版桌面程序可以安装到任何操作系统之上，包括 Windows、Linux、Unix、AIX 等。

单机版桌面程序包括.NET 和 JAVA 两个版本。在 DCenter 内核的基础上封装出来了.NET 程序集和 Jar 包程序，供用户开发 C/S 或 B/S 的程序。作为单机版产品，可直接调用.NET

程序集和 Jar 包来完成功能的研发。

3.2 B/S 开发框架

B/S 开发框架采用 Web 主流开发技术，为 WebGIS 的二次开发提供一整套解决方案。其 Web 开发框架的设计模式如下图所示。

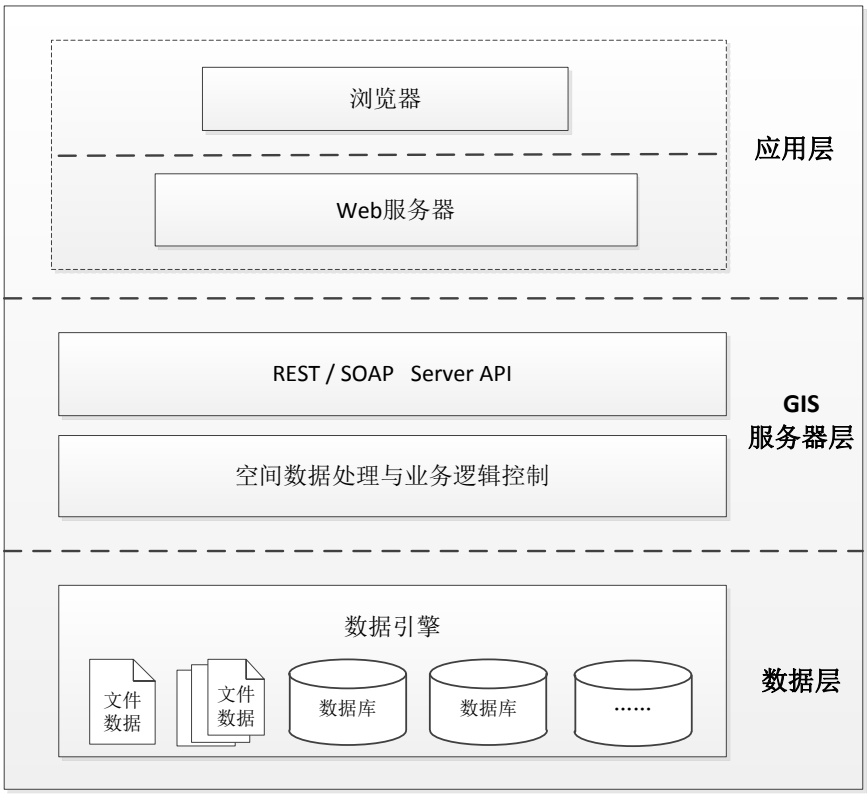


图 开发框架的设计模式

数据层：GIS 应用的数据源，包括本地数据源与网络数据源（Geodatabase），存储各种空间数据与非空间数据。采用 SDE 实现对空间数据的管理，异构数据源使用中间件，形成一套统一的数据管理机制。

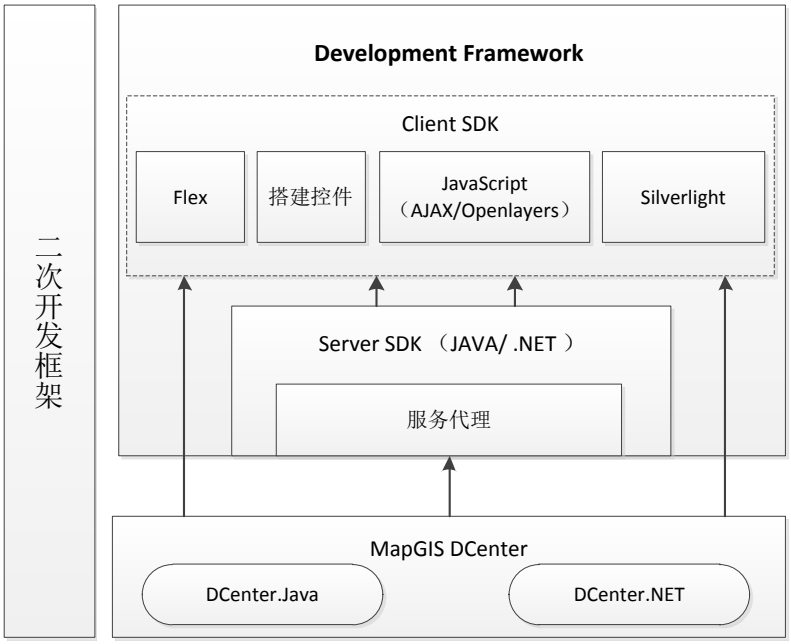
GIS 服务器层：基于数据中心的思想，底层提供数据仓库服务和功能仓库服务，采用 Web 服务的相关技术对基础内核服务进行封装，将复杂的 GIS 功能化繁为简，根据应用需求封装为粒度适中的 SOAP 与 REST 服务，提供统一的服务接口，实现空间数据存取与处理，完成业务逻辑运算等。采用功能仓库与工作流机制，空间分析等复杂功能采用工作流搭建形成功能模板，以服务方式提供。

应用层：根据应用模式，分为 Web 服务器与客户端。基于 Web 开发框架中的开发方式，

应用服务器端开发库进行开发，或者直接采用纯客户端开发方式调用服务接口完成处理。在执行访问应用程序时，Web 服务器端负责功能服务处理，进行网页的管理、业务流程控制；客户端浏览器作为视图层，通过通讯机制与服务端交互，在客户端生成用户界面、地图容器、查询结果及图表显示等。

基于上述设计思想、原则与设计模式，在平台的 Java 与.NET 的 GIS 服务器之上，构建统一的 Web 开发框架，为用户提供灵活、高效的二次开发解决方案，促进 WebGIS 在各行业的应用。B/S 的设计开发框架，其核心部分在二次开发层面，分别从服务器端与客户端提供相应的技术解决方案，即由相应的类库、控件、开发环境，以及应用示例与帮助手册组成的集合。

B/S 开发框架融合主流的 RIA 技术与特色的搭建开发机制，提供灵活的四大开发方式：Flex、Silverlight、JavaScript、搭建式，其 Web 开发框架的整体结构如下图**错误!未找到引用源。**所示。根据二次开发的应用需求，在服务端提供一致的 Java 与.NET 开发库，客户端均用插件与脚本技术。整套开发体系具有良好的兼容性，能够向下兼容。



设计开发框架体系架构

WEB 开发框架是一个有机组合体，基于 Web 服务端与客户端的各类 SDK，可独立或组合应用，具有较高的灵活性。根据常规应用模式，可分为纯客户端开发、服务器端开发与混

搭式开发三大类别。

3.2.1 纯客户端开发

纯客户端开发，即通过 Flex、Silverlight、JavaScript 等客户端开发技术进行 webGIS 的二次开发，采用那个纯客户端的开发方式，客户端直接与 GIS 服务通信，数据通信与业务逻辑均采用客户端技术实现。其中，Flex、Silverlight 的 RIA 开发与搭建式开发，功能集成到空间，非常简便易用。

Flex 开发：采用 Adobe 的 Flex 开发框架，与 WebGIS 平台的 GIS 服务体系紧密结合，提供独立的 Flex 开发方式，实现 GIS 在互联网的开发应用。

Silverlight 开发：与 Flex 类似，采用微软的 Silverlight 技术体系，将 GIS 应用与 WEB 前端技术紧密融合，提供基于 Silverlight 的 WEBGIS 开发方式。Silverlight 开发也是纯插件的客户端开发，无平台软件以来，开发简便，交互体验优秀。

AJAX/Openlayers 开发：基于 JavaScript 脚本技术与 AJAX 机制，结合平台的 GIS WEB 服务，采用 Openlayers 进行二次开发，调用规范的 GIS 服务实现地图数据的访问与其他 GIS 功能。基于 Openlayers 的脚本开发，无环境依赖，并提供很多友好的图形操作借口，开发简单易学。

搭建式开发：基于 MapGIS 提供的搭建机制，采用 JavaScript 脚本封装功能控件，并将其与 MapGIS 搭建平台结合起来，提供搭建式的 webgis 开发方式。搭建式的开发类似积木搭建方式，通过封装好的功能控件实现应用，对开发人员几乎没有门槛。

3.2.2 服务器端开发

服务器端开发主要是采用传统的 web 开发模式，基于 java 与.net 体系的平台开发库，与客户端的 javascript 脚本库进行结合的开发方式。为简化开发，平台的服务端对 GIS 服务接口进行封装，提供更易于用户应用的服务代理类 API，屏蔽复杂的过程，对外提供简单的开发接口。这种开发方式较为灵活，易于满足复杂的业务应用需求。

Java+javascript 开发：基于 java 开发体系，在服务端采用封装好的服务代理类库，即平台提供的 jar 包，结合客户端的 javascript 脚本库进行开发。

.net+JavaScript 开发：基于.net 开发体系，在服务端采用封装好的服务代理类库，即平台提供的 dll 库，结合客户端的 JavaScript 脚本库进行开发。

3.2.3 混合式开发

混合式开发主要指纯插件与客户端 JavaScript 脚本、或服务器段开发有机结合的一种开发模式。在不同的应用环境下，面对各种业务需求，可以通过平台的 web 开发框架，选用混合式开发实现。

插件与脚本混搭开发：即 flex/silverlight+javascript 开发模式。客户端插件与 JavaScript 混合开发，插件提供目录、地图、表格等容器，并在容器内提供数据的装载显示和用户交互功能，JavaScript 做为插件驱动，使用少量的脚本对插件进行控制，浏览器与 GIS 服务器通信在插件内部完成。

插件与服务器混搭开发：即 flex/silverlight+java/.net 开发模式，客户端插件与服务端结合的混搭开发。Flex 与 Silverlight 均可分别于 Java、.net 服务端开发结合，通过服务器端进行复杂的业务功能处理，在客户端插件中实现 GIS 功能的战士。