功能点分析基础

By David Longstreet
David@SoftwareMetrics.Com
www.SoftwareMetrics.Com
本文翻译：兰天
陕西新东方信息自动化有限公司

摘要

系统规模和复杂度不断增长，它们变得越来越难以理解。编码工具的改进使得软件开发者生产大量的软件以满足用户不断增长的需要。随着系统的增长，需要一种用来理解和交流系统规模的方法。功能点分析是一种能够解决这个问题的结构化技术。它是一种将系统分解为较小组件的方法，以便系统能够被理解和分析。

功能点是软件的度量单位，正如小时是时间的度量单位、公里是距离的度量单位、摄氏度是温度的度量单位一样。功能点度量这个单位同其它的度量单位如公里、华氏摄氏度、小时等等是一样的概念。

介绍

人类通过将问题分解为较小的可以理解的片断的方式来解决问题。看起来复杂的问题一旦被分解为较小的部分（被分成类）那么就会变得简单。对事务分类，将他们放在这个或者那个类别是一种惯用的做法。每个人都在日常生活中这样作。店主将货物按照货架存放，图书管理员将图书分目，秘书归档信件和文件。当被分类的对象是（软件）系统的内容时，就必须使用一套规则和定义来将这些对象放在合适的类别。即：分类方案。功能点是分类系统组件的一种结构化技术。他是一种将系统分解为较小组件的方法，以使系统能够更容易被理解和分析。它提供一种解决问题的结构化技术。

在功能点分析中，系统被分为５个大类组成部份（组件）和一些常规系统特性。前三类件是：**外部输入（External Inputs EI）、外部输出(External Outputs EO)和外部查询(External Inquiry EQ)**。这些组件中的每一个组件都处理档案，因此他们被称为“**事务**”**（transaction）**。另外两类或组件是：**内部逻辑档案(Internal Logical Files ILF’S)和外部接口档案（External Interface Files EIF’s）**，它们是构成逻辑信息的数据存储之地。系统常通用性评估系统的通用功能。

简史

功能点分析首先由Allan J. Albrecht（IBM的工程师）在１９世纪７０年代中期开发出来，它试图克服用代码行来估计系统规模所存在的难度，并且帮助开发出一种能够预期与软件开发相关的工作量的机制。这个方法首先在1979年公布，然后是１９８３年后期。１９８４年Albrecht精练了这个方法，并且从1986年国际功能点用户组织（International Function Point User Group）IFPUG成立之时开始，几个版本的《功能点计算实践手册》已经被IFPUG公布，当前版本的IFPUG手册是4.2版。全面的功能点培训手册可以从这个站点下载(**现在译者发现只有付会费的会员才可以下载**)。

功能点分析目标

通常情况下，术语－用户或者终端用户没有特定的含义，在这种情况下，它们指的是有经验的用户。这些人会从功能的角度来理解系统，有些更可能是提供需求和进行系统接受测试的人。因为功能点是从功能的角度来度量系统，因此它同所使用的技术无关。不考虑开发语言、开发方法、使用的硬件平台，因此系统的功能点数量会保持恒定，唯一的变化就是交付一套功能点所需要的工作量不同。因此，功能点分析能被用来确定是否组织之中或者组织之间所使用的工具、环境、语言有更高的生产率。这是功能点分析的关键点和伟大之处之一。

功能点提供了一种能够追踪和监控项目范围蠕变的机制。在需求、分析、设计、编码、测试和实现等各阶段结束之后的功能点数可进行比较。需求或设计结束之后的功能点数同实际发布产品的功能点数进行比较。如果项目增长了，那么就会存在范围蠕变。功能点增长的数量可以用来表明需求收集或者需求在项目组中交流的好坏程度。如果项目增长数量随时间下降则可以推断出同用户的交流已有所改进。

功能点分析质量特点

应该由受过培训或者有经验的人进行功能点分析。如果功能点分析由没有经验的人执行，则可以有理由认为功能点分析是不正确的。人工计算功能点应使用最新版本的《功能点计算实践手册》(Function Point Counting Practices Manaul) (最新是4.2)。

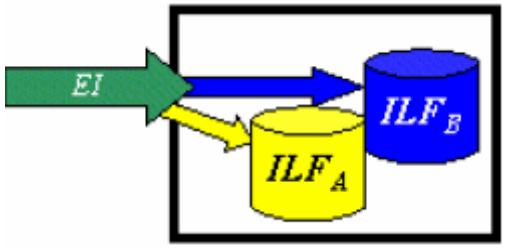
现有的应用文档应用来完成功能点计算。比如：界面格式、报表样式、系统间或者同其他系统的接口、逻辑或者初步的物理数据模型都将有助于功能点分析。

功能点分析的任务将作为项目整体计划的一部分。也就是说：功能点分析这个工作应有进度和计划。初步的功能点计算需要被开发以供规模评估之用。

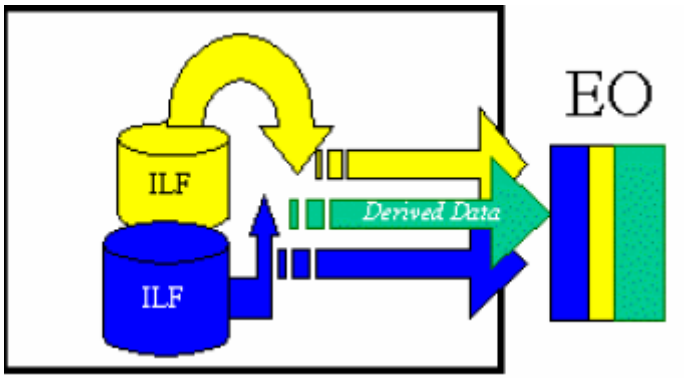
五个主要组件

对于计算机系统来说，同其他计算机系统交互是一个非常普遍的事情，因此，在分类组件之前必须划出每个被度量的系统的边界。必须要从用户的角度来划边界。简而言之，边界表明了被度量的系统或应用同外部系统或应用之间的界限。一旦边界被建立，则组件就能够被分类、分级和评分。

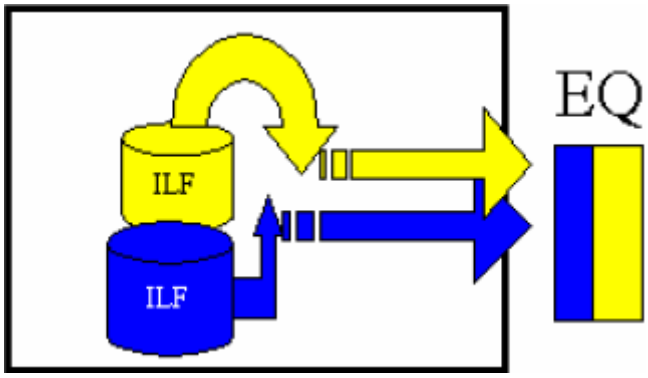
外部输入（External Inputs EI）-这是一个基本的过程，在这个过程中，数据穿越外部边界进入到系统内部。这里的数据可能来自于输入界面，也可以来自于另外的应用。数据将被用来维护一个或者多个内部逻辑档案(Internal Logical files)。数据既可能是控制信息，也可能是业务逻辑信息。如果数据是控制信息，则它不会更新内部逻辑档案。下图展现了一个更新两个FTR(File Type Referenced **FTR** 引用档案类型)的简单的EI。



外部输出(External Outputs EO)-这是一个基本的过程，在这个过程中，派生数据由内部穿越边界传送到外部。另外，一个EO可以更新ILF。数据生成报表或者传送给其他应用的数据档案。这些报表或者档案从一个或者多个内部逻辑档案以及外部接口档案生成。下图展现了一个EO和两个FTR，并且有从ILF派生出的派生信息（绿色）。



外部查询(External Inquiry EQ) – 一个基本过程，这个过程中的输入和输出部分都导致数据从一个或者多个内部逻辑档案或外部接口档案中提取出来。输入过程不能更新任何内部逻辑档案，并且输出端不能包括任何派生数据。下图展现了一个有两个ILF并且无派生数据的EQ。



内部逻辑档案(Internal Logical Files ILF'S) －用户可以识别的一组逻辑相关的数据，而且完全存在于应用的边界之内，并且通过外部输入维护。

外部接口档案（External Interface Files EIF's）－用户可以识别的一组逻辑相关数据，这组数据只能被引用。数据完全存在于应用的外部，并且由另一个应用维护。外部接口档案是另外一个应用的内部逻辑档案。

所有组件都被定级为高、中、低三个级别。

当组件被归为以上５类主要组件（EI's,EQ's ,EO's,ILF's,EIF's）中的一类之后，就要为之指定级别，对于事务组件（EI's,EQ's, EO's）来说，它们的级别取决于被更新或引用的档案个数以及**数据元素类型(Data Element Types DET's)**的个数。对于ILF's和EIF's档案来说，它们的级别取决于记**录元素类型（Record Element Types RET's）**和数据元素类型的个数。记录元素类型是ILF或者EIF中用户能够识别的数据元素小组。一个数据元素类型是一个用户可识别的、唯一性的、非递归的域。

下面的表格用来帮助定级过程（数字表示的级数在括号中）。例如：引用或者更新２个引用的档案类型（File Types Referenced FTR's）并且有７个数据元素的EI将被定级为中级，相关的级数是４。这里，FTR's 是被引用或更新的内部逻辑档案(ILF's)和被引用的外部接口档案(EIF's)的综合。

EI表(EI Table)

引用的文件类型个数（FTR’s）	数据元素(Data Elements)		
	1-4	5-15	>15
0-1	低	低	低
2	低	中	高
>=3	中	高	高

EO和EQ共用的表(Shared EO and EQ Tabgle)

引用的文件类型个数（FTR’s）	数据元素(Data Elements)		
	1-5	6-19	>19
0-1	低	低	中
2—3	低	中	高
>3	中	高	高

事务值(values of transactions)

级数(Rating)	值		
	EO	EQ	EI
低	4	3	3
中	5	4	4
高	7	6	6

如同所有的组件一样，EQ被评级和打分，基本上，EQ的定级同EO一样（低、中、高），但取值则同EI一样。级别取决于（综合了唯一输入和外端的）数据元素类型(DET’s)个数及被引用的文件类型(FTR’s)（综合了唯一的输入和输出端）的个数。如果同一个FTR被输入和输出同时使用，那么它只计算一次。如果同一个DET被输入和输出同时使用，那么它也只能被计算一次。

对于ILF’S和EIF’s来说，记录元素的行数量和数据元素类型个数来决定它们的低、中、高级别。记录元素类型(Record Element Type RET)是一个用户可以识别的ILF或者EIF中的数据元素小组。数据元素类型(Data Element Type DET)是ILF或者EIF中单一的、用户可识别的、非递归的域。

记录元素类型(RET’s)	数据元素(Data Elements)		
	1-19	20-50	>50
1	低	低	中
2-5	低	中	高
>5	中	高	高

级数(Rating)	值	
	ILF	EIF
低	7	5
中	10	7
高	15	10

将每个类型的组件的每一级复杂度计算值输入到下表中。每级组件的数量乘以所示的级数（numeric rating）得出定级的值（rated value ）.表中每一行的定级值相加得出每类组件的定级值之和。这些定级值之和再相加，得出全部组件的功能点值。这个和被填写到未调整的功能点数一栏中。

组件类型	组件复杂度(Complexity of Components)			
	低	中	高	全部
外部输入	____x3=____	____x4=____	____x6=____	
外部输出	____x4=____	____x5=____	____x7=____	
外部查询	____x3=____	____x4=____	____x6=____	
内部逻辑档案	____x7=____	____x10=____	____x15=____	
外部接口档案	____x5=____	____x7=____	____x10=____	
全部未调整的功能点数				
调整系数值				
全部调整后的功能点数				

调整系数值取决于１４个通用系统特性（General Sysytem Characteristics GSC's)。这些系统特性用来评定所计算功能点个数应用的通用功能的级别。每个特性有相关的描述以帮助确定这个系统特性的影响程度。影响程度的取值范围从０到５，从没有影响到有强烈影响。IFPUG的《功能点计算实践手册》提供了每个GSC's的评估准则。下表列出每个GSC的概要。

通用特性		描述
1.	Data communications	How many communication facilities are there to aid in the transfer or exchange of information with the application or system?
	数据通信	多少个通信设施在应用或系统之间辅助传输和交换信息。
2.	Distributed data processing	How are distributed data and processing functions handled?
	分布数据处理	分布的数据和过程函数如何处理？
3.	Performance	Was response time or throughput required by the user?
	性能	用户要求相应时间或者吞吐量吗？
4.	Heavily used configuration	How heavily used is the current hardware platform where the application will be executed?
	硬件负荷	应用运行在的硬件平台工作强度如何？
5.	Transaction rate	How frequently are transactions executed daily, weekly, monthly, etc.?
	事务频度	事务执行的频率(天、周、月)如何？
6.	On-Line data entry	What percentage of the information is entered On-Line?
	在线数据输入	在线数据输入率是多少？
7.	End-user efficiency	Was the application designed for end-user efficiency?
	终端用户效率	应用程序设计考虑到终端用户的效率吗？
8.	On-Line update	How many ILF's are updated by On-Line transaction?
	在线更新	多少ILF被在线事务所更新？

9.	Complex processing	Does the application have extensive logical or mathematical processing?
	处理复杂度	应用有很多的逻辑或者数据处理吗 ？
10.	Reusability	Was the application developed to meet one or many user's needs?
	重用性	被开发的应用要满足一个或者多个用户需要吗？
11.	Installation ease	How difficult is conversion and installation?
	易安装性	升级或者安装的难度如何？
12.	Operational ease	How effective and/or automated are start-up, back-up, and recovery procedures?
	易操作性	启动、备份、恢复过程的效率和自动化程度如何？
13.	Multiple sites	Was the application specifically designed, developed, and supported to be installed at multiple sites for multiple organizations?
	跨平台性	应用被设计、开发和支持被安装在多个组织的多个安装点（不同的安装点的软硬件平台环境不同）吗？
14.	Facilitate change	Was the application specifically designed, developed, and supported to facilitate change?
	可扩展性	应用被设计、开发以适应变化吗？

关于GUI应用的考虑

诸如事务频度、终端用户效率、在线更新、可复用性等GSC项通常在GUI应用中的评分要比传统应用要高。另一方面，性能、硬件负载、跨平台性等BSC项在GUI应用评分比传统应用要低。

一旦所有１４个GSC项的评分值都被确定下来，则通过IFPUG的值调整公式可以计算出调整系数值（VAF）

$$VAF=0.65+[\sum_{i=1}^{14} Ci/100\,]$$

这里：

VAF:值调整系数（Value Adjusted Factor）

Ci＝每个通用系统特性的的影响程度。

i=从１到１４代表每个GSC。

Σ=所有１４项GSC的和。

另一种容易理解的公式是：VAF=(65+TDI)/100,这里TDI是所有１４项GSC影响度的和。

功能点分析优点总结

- 能被用来精确度量软件规模。规模是确定软件生产率的一项重要内容。
- 可被用来作为度量和管理范围蠕变的基本成分。
- 是创建评估模型的基础，这个模型能被解释、修订和精确。
- 能被用于其他度量中，以帮助精确定位改进时机。
- 能有助于改进同高级管理者的交流。

- 能被不同的人、在不同的时间计算，可在合理误差范围内获得相同度量结果。
- 能使不懂技术的用户容易理解，这有助于同客户或者消费者交流系统的规模。
- 能通过对工具、语言、环境的比较来确定哪个更有生产效率。

结论

准确计算软件规模已经困扰了软件业长达４５年之久，功能点作为度量软件规模的标准度量法则已经被越来越广泛地接受。现在功能点已经使精确的规模度量变为可能。现在可以预期到的是：软件质量和软件生产率将被全面提升。了解软件规模是了解软件产生率的关键。没有可靠的规模度量方法，则相关的生产率（功能点数/每月）变化或者相关的质量（缺陷/功能点）的变化就不能被统计出来。如果相关的生产率和质量的变化能随时统计和策划，则组织可以将注意力集中到组织的强项和弱项上。更为重要的是：任何试图改进弱项的努力都可以被度量以确定其效果。

---

www.SoftwareMetrics.Com
David@SoftwareMetrics.com
Longstreet Consulting inc
2207 S. West Walnut
Blue Springs ,MO 64015
(816) 739-5058