Panorama 系统程序开发规范之二

1.匈牙利命名规则

变动	前缀	类型			
	a	Array			
	b	Boolean			
	by	Byte			
	c	Char	//有符号型字符		
	cb	Char Byte	//无符号型字符（没多大用处）		
	cr	ColorRef	//颜色参考值		
	cx,cy	Length of x,y (ShortInt)	//坐标差（长度）		
	dw	Double Word			
	fn	Function			
	h	Handle			
	i	Integer			
	m_	Member of a class			
	n	Short Integer			
	np	Near Pointer			
	p	Pointer		lp	Long Pointer
×(str)	s	String			
	sz	String with Zero End	//以字符'\0'结尾的字符串		
	tm	Text	//文本内容		
	w	Word			
	x,y	Coordinate	//坐标		

2.Panorama 系统的命名约定

2.1 VC 中变量命名时的前缀约定

Array	a...	//例：CStringArray saText
BOOL	b...	
UINT	n...	
int	i...	
short	n...	
long	l...	
WORD	w...	
DWORD	dw...	
float	f...	
char	c...	
char*	psz...	
TCHAR*	psz...	
LPCTSTR	lpsz...	
CString	str...	
COLORREF	cr...	
LPLOGPALETTE	lp...	(包括 LP 开头的类型都是这样)
POINT	pt...	
CPoint	pt...	
HANDLE	h...	
HGLOBAL	h...	(包括 H 开头的类型都是这样)

说明：

1.如果是指向上述类型的指针，就在上面规范前加 "p" ;

2.如果是指向上述类型的双重指针，就在上面规范前加 "pp" ;

3.如果是类成员变量，则在上面规范前加 "m_";

4.全局变量,则在上面规范前加 "g_";

5.在类型前加了"const",命名约定不变;

2.2 VC 中变量命名时的后缀约定

1.MFC 类

CWnd* p...Wnd 省去的地方一般为该类的用途 (如果是某一个类的成员，则还应该在前加 "m_")又如：CView* p...View

2.3 局部变量应尽量易懂简洁，使用常见的变量，如 Num,nCount,i,j,k,n,len,pos, offset,nReadNum,index,nRet,ret, string,filename 临时变量，如 ltmp,ftmp,tmpStr,tempStr 。。。

2.4 函数命名也应该见名知意。如 CalcAllDataStyle(),ReadDocDataFromTime(),GetIndexInfo()

常见的函数 Init_, Open_, Create_, Get_, Set_, Read_, Load_, Write_, Start_, Stop_, Check_, Test_, Fill_, Process_, Sort_, Do_, Select_, Is_, Exist_,_Ex…

2.5 禁止使用汉语拼音来命名；

2.6 在代码中尽量不用具体的大小数值，定义成宏，便于以后维护,如：
#define MAX_DOWNLOADNUM 20
struct DownInfo m_DownInfo[MAX_DOWNLOADNUM];

2.7 VC 中一些控件的缩写：

ComboBox	cmb
Edit	edt
Dialog	dlg
ListBox	lst
Picture	pic
Animate	ani

3. 编排

3.1 函数间要有空行分开，一个程序中的空行数目最好占 8%-16%；

3.2 变量的定义尽可能放在最开始处，多态函数和功能相近的函数集中放在一起;

3.3 声明变量时对齐变量名,并在定义时加以注释说明；

4. 程序开发环境规约

4.1 工作目录结构的规定：

```
project name(项目名称)
——bin        执行文件
——log        日志文件
——lib        库文件
——include    头文件
——src        源程序
——dat        数据文件
```

4.2 工程中不起作用的文件或类应删除，工程目录下的非工程文件也应该移走，保持工程的清洁，避免混淆难于管理;

4.3 在 VC 环境下，建议将常用的头文件全部放入 stdafx.h 中，而在每个 cpp 开始处嵌入 stdafx.h。避免头文件的交叉引用，如果有严重的交叉引用，适当使用类的声明。

4.4 将独立性比较强的模块抽出来，做成 DLL，控件或 COM 组件，该模块可单独编写和测试，也增强了其可重用性。

4.5 一个比较大的工程应留有一定的消息接口或插件接口等。

4.6 工程的版本控制要严格，版本格式为 xx.xx.xx，必要时使用 Build 次数或日期。高版本尽量兼容低版本的用法、数据或协议。

4.7 工程的编译宏定义和工程参数设置应正确，每作一个新工程时应检查工程参数是否正确。建议字节对齐方式为 1 字节对齐。

5. 程序备份

5.1. 要有备份记录

备份时注明备份日期和主要增加的功能

5.2. 定时备份

根据程序量的多少，可以每天备份一次，也可以半天备份。

5.3. 多种介质备份

至少在硬盘上做 2 个备份，在软盘上做一个备份；在使用他人主机进行备份时，不可放于没有密码保护的 ftp 服务器上，可以发送到自己的 email 信箱中进行备份。

5.4 在软盘上备份时，应该去掉中间文件和执行文件，vc 可以自动生成的文件如*.clw, *.ncb, *.opt 等也可删除，最后压成一个 zip 文件，复制到软盘中。

6. VC 程序界面设计规范

6.1.颜色选择：

6.1.1 基调应以 WINDOWS 颜色（灰色）为主，同一个窗体中除白、黑、灰色之外，其它的颜色总数不宜超过 3 种（对以生产和学习为目的的软件而言，娱乐性软件可以做得花哨一些；

6.1.2 窗体和控件（除 EDIT、RICH EDIT 等编辑控件外）的背景色也宜用灰色，当一个控件有输入焦点或鼠标焦点时，可以用较明亮的颜色;

6.1.3 非激活状态下，字体前景宜用黑色，背景用灰色；

6.2.字体的选择：

6.2.1 汉字字体一般选宋体，字体大小选 10 号；

6.2.2 一般选系统常用的字体，绝对不要选自己加入的而系统没有的字体；

6.3.图片的选择

6.3.1 在窗体的工具拦中的按钮可以用图标，文字可以写上也可以不写，如果不写则一定要使用 tip 来提示用户该按钮完成的功能；

6.3.2 在按钮中使用的图片要能表达按钮对应功能的意义；

6.3.3 不宜对普通的按钮只用图标做外观；

6.4.操作的便利性

6.4.1 为方便用户的使用，所有的输入控件应该按 tab 键和回车键排序，特别是密码输入时，应该能用回车切换输入框的焦点；

6.5.数据安全

6.5.1 对程序的退出、写数据等有破坏可能或数据丢失可能的操作应该给用户一次确认的机会；

6.6.帮助文件

6.6.1 帮助文件宜用 html 格式，因为 hlp 格式的文件只能在 windows 中用；

6.7.窗体大小的确定

6.7.1 一般窗体的大小应该可以让用户自己调整，窗体的初始长宽比例为 4：3

6.7.2 要考虑到用户可能会用到不同的分辨率，在开发时应使用当时流行的分辨率；

6.7.3 除非必要，否则不宜用模态窗体,但可以让用户选择使窗体成为模态窗体；

6.7.4 应用程序的大小不固定时(拖动窗口的右下角时可以改变窗体大小)，应处理窗体变化时窗体内各控件大小和位置的变化；

6.8.视图的选择(单/多文档界面类型)

6.8.1 对于一个简单的文本编辑器应用程序，选择 CEditView;

6.8.2 对于一个能编辑多信息文本格式(RT F)文件的应用程序，选择 CRichEditView(这一选择将导致应用程序为文档类选择 CRichEditDoc 类);

6.8.3 对于一个图形应用程序，选择 CScrollView;

6.8.4 对于一个简单的监控或帐目管理应用程序，选择 CListView;

6.8.5 要着手创建一个资源管理器类型的应用程序，请选择 CTreeView(在以后的步骤中,可以手工添加一个
CListView);

6.8.6 在对话框模板外创建一个视图，选择 CFormView(一个对话框是一个被几个控件窗口占据的窗口，诸如按钮和编辑框);

6.9. MFC 应用程序类型的选择：

6.9.1 如果创建一个用户界面需求有限的应用程序，或如果想界面完全单一，那么就创建一个对话框应用程序。典型的对话框应用程序包括配置硬件设备的应用程序、屏幕保护程序和游戏程序等;

对话框要易用且简洁，字体和控件的组织搭配要得体，能简单不复杂，各控件的焦点、Tab 顺序等要讲究，视应用场合要适当支持键盘。在简洁易用的前提下，力求个性化，设计

得更加友好。程序各对话框的风格要保持一致。

6.9.2 如果应用程序要编辑一个文档，应该选择单/多文档界面类型。这里的“编辑一个文档”是广义上的意思，所指的文档可以是一个文本文件、电子数据表文件、第三方数据库的一个或多个表、或者是自己的二进制文件，甚至可以是大量硬件设备的储存设置。编辑仅仅表示对其中任何一个类型的文档进行添加、删除或修改操作。

6.9.3 单文档界面应用程序一次只允许处理一个文档。如果应用程序实际上一次只需处理一个文档，诸如监视一组硬件设备的应用程序，那么应该选择单文档界面；否则应该创建一个多文档界面应用程序，即使在开始时一次编辑多个文档并未显出有任何好处。

6.9.4 一个多文档界面应用程序允许一次编辑多个文档，它并不比一个单文档界面应用程序复杂，但却带来了一次至少查看多个文档的方便。

6.9.5 在重要的窗口或区域应能弹出右键，实现常见操作。工具栏上放最常用的操作按钮，必要时动态更换按钮。状态栏显示足够多的有用信息。消息主控在 Mainframe 中，单文档的主控也可在 View 中，所有的对话框的弹出或非模态对话框的控制都在主控窗口中完成，具体的数据处理放在单独的文件中或设计成类。在 App 类中实现 Ini 读写，各数据对象的定义和析构，全局变量的赋值和初始计算，存盘退出等。各视图的 OnDraw 和 GDI 画图尽量使用内存位图的方式，以免闪烁。

6.10. 操作进度指示

6.10.1 把鼠标光标暂时变成沙漏形状，以指示一个漫长的操作，要求用户应该等待。

6.10.2 可以用沙漏光标指示短暂的等待。对于长时间的等待，可以考虑使用一个无模式对话框，并在上面显示简短的消息，描述正进行什么处理;

6.11. 分隔线控件

6.11.1 为统一起见不要使用分组框、按钮等控件做分隔线，应按如下做法：用 Picture Control，属性设为 Etched 和 Frame，使该控件缩小到一条直线；

7. 其他

7.1. 为保证系统间的兼容性,不使用 int 类型(因为不同系统之间的存储字节长度往往不同),应使用 long 或 short 型。

7.2. 头文件名应小写,如用#include “abcdef.h”;

7.3. 本系统中注释统一只用“//”;

7.4 if(0 == GetDataType(…))比 if(GetDataType(…) == 0) 好，纵使误将==写成=，在编译一层就会报错。

7.5 函数定义 short GetInputType(const char * lpzInput)比 short GetInputType (char * lpzInput) 好，以免 lpzInput 在函数体中被破坏。

7.6 变量在定义时赋初值，类析构时或程序退出时判断释放所有变量。

7.7 编码空间一定要充分预留，编码时注意可扩充性, 如：定义保留字段，供以后扩充使用

7.8 不要大量使用无符号型变量。无符号变量在判断时易造成错误，甚至死循环，尽量少用。

7.9 少使用 malloc,free，realloc；多用 new,delete；new,delete 是规范的 C++语法，通用性强，

realloc 易造成内存抖动。

7.10 代码中不要用"+2","+4",要用"+sizeof(short)","+sizeof(int)"；不要用 filename[40],而是filename[MAX_PATH]。

附录 A 程序维护手册格式说明

文档编号
版 本 号
密　　级

文档名称　　XXXX 程序维护手册

项目编号：
项目名称：
开发部门：
项目负责人：

编写　　年　月　日
校对　　年　月　日
审核　　年　月　日
批准　　年　月 日

程序维护手册
1 引言
1．1　编写目的
　　［ 阐明编写维护手册的目的，简述其内容。指出读者对象（程序维护人员、研发人员）。］

1．2　开发单位
　　［说明项目的提出者、项目的委托单位、开发单位和使用场所。］

1．3　定义
　　［ 列出本文挡中用到的专业术语的定义和缩写词的原文。］

1．4　参考资料
　　［ 可包括：a.用户操作手册；b.于本项目有关的文档。列出这些资料的作者、标题、编号、发表日期、出版单位或资料来源以及保密级别。］

2 系统说明

2．1　系统用途

［ 说明系统具备的功能，输入和输出。］

2．2　安全保密

［ 说明系统安全保密方面的考虑。］

2．3　总体说明

［ 说明系统的总体功能、对子系统和作业作出综合性的介绍，并用图表方式给出系统主要部分的内部关系。］

2．4　程序说明

［ 说明系统中每一程序、分程序的细节和特性。］

2．4．1 程序 1 的说明

2．4．1．1　功能［ 说明程序的功能。］

2．4．1．2　方法［ 说明实现方法。］

2．4．1．3　输入［ 说明程序的输入、媒体、运行数据记录、运行开始时使用的输入数据的类型和存放单元、与程序初始化有关的入口要求。］

2．4．1．4　处理［ 处理特点和目的，如：a. 用图表说明程序的运行的逻辑流程；b. 程序主要转移条件；c. 对程序的约束条件；d. 程序结束时的出口要求；e.与下一个程序的通信与联结（运行、控制）；f. 由该程序产生并供处理使用的输出数据类型和存放单元。g. 程序运行所用存储量、类型及存储位置等。］

2．4．1．5 输出［ 程序的输出。］

2．4．1．6　接口［ 本程序与本系统其他部分的接口。］

2．4．1．7　表格［ 说明程序内部的各种表、项的细节和特性。对每张表的说明至少包括: a. 表的标识符；b. 使用目的；c. 使用此表的其他程序；d. 逻辑划分，如块或部，不包括表项；e. 表的基本结构；f. 设计安排，包括表的控制信息。表目结构细节、使用中的特有性质及各表项的标识、位置、用途、类型、编码表示。］

2．4．1．8　特有的运行性质［ 说明在用户操作手册中没有提到的运行性质。］

2．4．2 程序 2 的说明［ 与程序 1 的说明相同。以后其他各程序的说明相同。］

3 操作环境

3．1　设备

［ 逐步说明系统的设备配置极其特性 ］

3．2　支持文件

［ 列出系统使用的支持软件、包括他们的名称和版本号。］

3．3　数据库

［ 说明每个数据库的性质和内容，包括安全考虑。］

3．3．1 总体特征

［ 如：a. 标识符 b. 使用这些数据库的程序；c. 静态数据；d. 动态数据；e. 数据库的存储媒体；f. 程序使用数据库的限制。］

3．3．2 结构及详细说明
3．3．2．1　说明该数据库的结构，包括其中的记录和项；
3．3．2．2　说明记录的组成，包括首部或或控制段、记录体；
3．3．2．3　说明每个记录结构的字段，包括：标记或标号、字段的字符长度和位数该字段的允许值范围。
3．3．2．4　扩充：说明为记录追加字段的规定；

4 维护过程
4．1　约定
[列出该软件系统设计中所使用全部规则和约定，包括：a. 程序、分程序、记录、字段和存储区的标识或标号助记符的使用规则；b. 图表的处理标准、卡片的连接顺序、语句和记号中使用的缩写、出现在图表中的符号名；c. 使用软件的技术标准；d. 标准化的数据元素极其特征。]

4．2　验证过程
[说明一个程序修改后，对其进行验证的要求和过程（包括测试程序和数据）及程序周期性验证的过程。]

4．3　出错及纠正方法
[列出出错状态及其纠正方法。]

4．4　专门维护过程
[说明文档其他地方没有提到的专门维护过程，如：a. 维护该软件系统的输入部分（如数据库）的要求、过程和验证方法；b. 运行程序库维护系统所必须的要求、过程和验证方法；c. 对闰年、世纪变更所需要的临时性修改等。]

4．5　专用维护程序
[列出维护软件系统使用的后备技术和专用程序（如文件恢复程序、淘汰过时文件的程序等）的目录，并加以说明，内容包括：a. 维护作业的输入输出要求；b. 输入的详细过程及硬件设备上建立、运行并完成维护作业的操作步骤。]

4．6　程序清单和流程图
[引用资料或提供附录给出程序清单和流程图。]

附录 B　部分编程常用单词缩写

规则：较短的单词可通过去掉“元音”形成缩写；较长的单词可取单词的头几个字母形成缩写；一些单词有大家公认的缩写.

完整单词	可缩写为	缩写
A		
average	------>	avg ;

B
back ------> bk ;
background ------> bg ;
break ------> brk ;
buffer ------> buf ;

C
color ------> cr ; (clr)
control ------> ctrl ;

D
data ------> dat ;
delete ------> del ;
document ------> doc ;

E
edit ------> edt ;
error ------> err ;
escape ------> esc ;

F
flag ------> flg ;
form ------> frm ;

G
grid ------> grd ;

I
increment ------> inc ;
information -----> info ;
initial -----> init ;
insert ------> ins ;
image ------> img ;

L
label ------> lab ;
length ------> len ;
list ------> lst ;
library ------> lib ;

M
manager ------> mngr ; (mgr)
message ------> msg ;

O

Oracle ------> Ora ;

P

panorama ------> pano ;
password ------> pwd ;
picture ------> pic ;
point ------> pt ;
position ------> pos ;
print ------> prn ;
program ------> prg ;

S

server ------> srv ;
source ------> src ;
statistic ------> stat ;
string ------> str ;
Sybase ------> Syb ;

T

temp ------> tmp ;
text ------> txt ;

U

user ------> usr ;

W

window ------> wnd ; (win)