

类 C 脚本架构设计及其在电力系统中的应用

陈宏君,熊 蕙

(南京南瑞继保电气有限公司,江苏 南京 211102)

摘 要:不同国家和地区的用户在电力系统装置功能和配置上有一定的差别,要求基于工具软件实现装置的灵活配置需求。研发人员在采用模块化、图形化元件搭建装置功能过程中,也需要有手段减轻人机接口配置工作量。为此,提出类 C 脚本的解决方案,设计了适用于电力系统装置配置建模的 API 接口函数,支持设置变量属性、信号连线、定值分组、元件投退、IEC61850 建模等功能。开发了脚本解析器,通过构建基于表驱动的词法提取、基于命令模式的脚本执行、基于递归下降法的表达式计算、基于多级 Hash 查找等关键算法,实现了脚本快速解析执行,6 万行脚本解析耗时小于 3 s。基于该脚本架构,装置研发人员可定义模块化元件脚本,实现多个元件实例配置的自动化复用;可定义全局脚本,根据用户选项,触发条件执行语句,实现灵活的装置选型配置。实践表明,类 C 脚本显著提高了电力系统保护测控装置的研发效率和产品的适用范围。

关键词:类 C 脚本;元件脚本;全局脚本;装置配置

中图分类号:TP39

文献标识码:A

文章编号:1673-629X(2017)03-0171-05

doi:10.3969/j.issn.1673-629X.2017.03.036

Design of C Style Script Structure and Its Application in Power System

CHEN Hong-jun, XIONG Hui

(NR Electric Co., Ltd., Nanjing 211102, China)

Abstract: In different countries and regions, power system users have certain difference on the device function and configuration, so the tool software need to achieve the flexible allocation of equipment requirements. Developers also need to reduce the HMI configuration work when using modular and graphical components to setup the device. The solution of C style script is presented, in which API interface is suitable for power system device configuration, and the scripts support variable properties setting, signals linking, parameters grouping, enable or disable components, IEC61850 modeling and other functions. The main modules and script parser processes are implemented by establishment of the key algorithms including lexical extraction based on table driven, script execution based on command mode, expression calculation based on the recursive descent method, and lookup based on multiple Hash. 60 000 lines of resolution takes less than 3 seconds. Device developers can define components script, to realize the automatic configuration of reuse. They also can define global scripts, which provide maximum device features and configuration options. Practice shows that class C script can significantly improve the development efficiency of measurement and control device for power system protection and scope of the product.

Key words: C style script; component script; global script; device configuration

0 引言

电力系统控制保护产品面向平台化、智能化的方向发展,要求设计和开发通用硬件、软件平台,以提高装置研发的可靠性和竞争力^[1-5]。在面向国内外的工程实施中,不同的用户往往有不同的需求。国际市场用户要求支持装置选型、系统配置、功能投退等可配置功能,要求可选择装置软件的版本、应用场景,例如选择是分段单 CT、分段双 CT 或母联双 CT 等场景;要求

可配置间隔数目、母线电压输入使能、9-2 组网模式等。如果每个工程都进行定制化开发,开发周期长,维护成本高。需要提供一种平台化的产品适应性开发方法,满足用户二次配置需求。可视化编程配置逐渐成为主流研发模式,可基于模块化元件多次实例实现保护测控功能的复用^[6]。在完成装置功能的可视化编程配置后,还需要进行装置人机 HMI 接口的配置,例如配置告警变位事件、配置模拟量上送分组、配置

收稿日期:2016-05-03

修回日期:2016-09-07

网络出版时间:2017-02-17

基金项目:国家“863”高技术发展计划项目(2015AA050101)

作者简介:陈宏君(1981-),男,硕士,高级工程师,研究方向为电力系统平台软件开发。

网络出版地址: <http://www.cnki.net/kcms/detail/61.1450.TP.20170217.1632.074.html>

IEC61850 模型等。这些工作通常是装置研发集成人员手工操作,也需要提供方法,在制作模块化符号元件时可以进行元件相关的 HMI 设置,实现 HMI 配置的复用。

脚本是依据一定的格式编写的文本,由程序解释执行,具有易于掌握、灵活扩展等优点^[7-8]。文献[7]定义了嵌入式 Web 服务器脚本解析的主要接口,设计了脚本引擎的调用方法。文献[8]提出一种自定义脚本的实现方法,设计了用于描述界面布局和参数格式的脚本语句,实现系统界面的灵活配置。文献[9]提出一种基于模型的自动判断方法,利用脚本驱动判断模型执行对测试数据的判断过程,提高判读效率和准确度。文献[10]研究了语义可配置的模型转换技术,利用基于 OCL 的脚本语言(TSS)来描述转换语言的语义。

为提高模块化配置效率和实现面向用户的选型配置功能,文中设计了类 C 脚本,支持在模块化元件研发中编写元件脚本、装置集成研发过程中编写全局脚本。类 C 脚本支持 if-else 等控制语句,支持四则运算、标准 C 库函数、自定义 API 函数。

1 类 C 脚本设计

1.1 脚本需求分析

- 采用可扩展的脚本系统,源自如下关键需求:
- (1)实现配置过程的自动化和多个元件实例配置的重用;提高模块集成效率。例如支持将信号添加到装置菜单分组中,将定值添加到定值分组中,把变量填写到 IEC61850^[11-12]逻辑节点对应的数据属性中。
 - (2)根据全局配置选项,条件使能执行相关功能,自动修改对应设置,实现 1 个工程支持多场景模式开发维护。
 - (3)一些特殊需求通过专属脚本实现,减少对工具界面编辑的二次开发及对其他应用的影响。

1.2 脚本函数设计

脚本设计要求简单,易于研发人员掌握,支持灵活组合,适用于电力系统保护测控装置的配置场景和满足用户配置需求。脚本由一系列经过精心设计的 API 函数和控制语句组成。按照使用范围,可分为全局脚本和元件脚本。全局脚本是装置集成研发人员在装置选型和界面定义的控制文本段,是在脚本解析时优先执行的函数文本。元件脚本是模块研发人员在元件图形库制作阶段定义的配置文本段。脚本函数定义的通用形式如下:

- (1)返回值类型 函数名(形参 1,形参 2,形参...);
- (2)返回值类型 函数名(条件执行表达式,形参 1,形参 2,形参数据

脚本中支持 if、if-elseif-else、if-else 等条件控制段,例如:

```
if(表达式){
脚本函数段 1...
}

else if(表达式){
脚本函数 2...
}

else{
脚本函数 3...
}
```

对脚本功能按照大类进行归类精简,规范统一函数命名,通过传入不同形参细分子选项。通过对电力系统装置开发配置场景进行规划,共定义了 50 余个脚本 API 接口。按照功能可分为:宏定义、信号拉线、信号描述设置、信号属性设置、引用表操作、定值分组和操作、页面和元件使能操作、IEC61850 通信中 CID(装置能力描述)建模、设置插件型号、修改元件执行顺序等。表 1 给出了典型常用的脚本函数和功能定义。

表 1 脚本函数定义

函数名	功能
SetDesc	设置变量描述
DefMacro	宏定义
AddLink	信号拉线
SetAttr	设置变量的属性
AddTab	将变量添加到引用表
AddParaGroup	将变量添加到定值组
Enable	使能页面、元件
Disable	退出页面、元件

- 以信号描述设置为例,脚本设计如下:
- (1)功能设计:设置变量和定值中文、英文描述。
 - (2)接口定义:bool SetDesc(char * pvar, char * desctype, char * pdesc)。
 - (3)形参说明:pvar 变量名;desctype:描述类型;pdesc:描述值。
 - (4)使用示例:SetDesc(IA, cn, "@ B01. Bus1 @ 母差电流 A 相")。

2 脚本解析设计

脚本函数是面向应用设计的接口服务,解析执行应用定义的脚本段。脚本的处理可分解释型执行、编译型执行。解释型执行如 MediniQVT,通过编码的方式实现转换语言的执行语义^[13],编译型执行工具如 ATL,将脚本编译成字节码,然后在虚拟机上执行^[14]。文中采用解释执行的策略,脚本执行的流程包括词法分析、if 语句分支执行、根据脚本类型调用解析函数。

解析单个函数时进行表达式计算、形参处理、调用装置配置处理软件开放的服务接口,例如变量查找、属性设置等接口。脚本解析可划分为词法提取模块、脚本执行模块、表达式计算模块。

2.1 词法提取模块

词法提取模块,用于读取脚本段,并进行标记分类和管理,其关键实现点是基于表驱动的字符串查找表和前向探测单词匹配技术。词法提取模块建立 ASCII 字符查找表,可快速返回每个字符对应的类型。定义单个字符类型如下:

```
enum CharacterType{
    BLANK=01, //空白
    NEWLINE=02, //新行
    LETTER=04, //字母
    DIGIT=010, //数字
    HEX=020, //HEX
    OTHER=040, //其他
};
```

预先对 ASCII 字符编码进行分类,构建字符的 ASCII 码对应字符类型查找表:

```
static unsigned char map[256]={
    0, //000 nul
    BLANK, //011 ht
    NEWLINE, //012 nl
    ...
    BLANK, //040 sp
    DIGIT, //060 0
    LETTER|HEX, //101 A
    ...
    LETTER, //132 Z
    OTHER, //176 ~
};
```

则通过 map[unicode]可快速返回该字符对应的类型。通过对各个字符和后续字符的类型探测组合判断,可提取出各个单词。单词 CToken、词法扫描类 CLex 的主要接口和属性定义如下:

```
//单词标记类
class EXPR_DECLSPEC CToken{
public:
    CToken();
    CToken( const Qstring& s,int mtp=0,intstp=0);
    virtual ~CToken();
public:
    int m_mainntp; //token 主类型
    int m_subntp; //子类型
    QString m_str; //名字、值
};

//词法扫描类
class EXPR_DECLSPEC CLex {
```

```
public:
    CLex( const Qstring& text);
    virtual ~ULex();
public:
    int count() {return m_tklist.count();}
    CToken * at(int i) {return m_tklist.at(i);}
    void lexParse( Qstring& text,int len);
protected:
    QList<CToken * >m_tklist;
    int m_pos;
    int m_len;
    QString m_text;
}
```

定义每个单词 Token 的主类型定义,如 KEY-WORD(关键字)、IDENTIFIER(标识符)、CONSTANT(常量)、STRING(字符串)、OPERATOR(运算符)、PUNCTUATOR(分隔符)、PREPROCESSOR(预处理)。每种主类型还细分子类型。以运算符为例,还细分如加、减、乘、除等各种子类型,例如 COMMA(" , ")、COLON(" : ")、PLUS(" + ")、MINUS(" - ")、MULT(" * ")、DIV(" / ")等。词法扫描的主算法见图 1。

输入: text: 待解析的脚本文本
len: 文本长度输出:
输出: 无

void CLex::lexParse(QString& text, int len) {
 m_text = text; m_len = len;
 定义:
 int curpos = 0; //扫描文本内容的游标
 QChar ch(0); //当前文本的字符内容
 Ushort chcode = 0; //ch对应unicode编码

do {
 ch = text.at(curpos);
 chcode = ch.unicode();
 ++curpos;
 switch(chcode) {
 //1: a-z, A-Z 可能属于变量命名的类型
 case 'a': case 'Z': case '_':
 parseIdentifier(curpos, chcode); break;

 //2: 数字0-9可能属于数值常量类型
 case '0': case '9':
 parseDigit(curpos, chcode); break;

 //3 提取分隔符
 case ' ': case '\t': case '\n': case '\r': case ':': case '\\':
 parsePunctuator(curpos, chcode); break;

 //4: C语言运算符: () [] < > + - * / ! & 等
 case '!': case ':': case '(': case ')': case '[':
 case '+': case '-': case '*': case '/':
 case '!': case '&': case '|': case '^':
 parseOperator(curpos, chcode); break;

 //5 提取常量字符串
 case '\"': case '\"':
 parseConst(curpos, chcode); break;

 //处理预编译字符
 case '#':
 parsePreprocessor(curpos, chcode); break;
 } **while**(curpos< len) }

图 1 词法扫描主算法

在主算法中,根据当前字符的类型,如果是[a-z]、[A-Z]、‘_’,则是变量命名或关键字的首字母,进入 parseIdentifier 单词标记提取函数,如果是数字、运

算符、常量、预编译字符,则进入相应的提取子函数。其中单词提取是重要的步骤,其算法流程如图 2 所示。

```
//解析标记符,同时提取关键字
//m_text:待解析的文本 m_len:文本长度
void CLex::parseIdentifier(int &curpos,
Ushort chcode){
switch(chcode) {
//尝试匹配auto关键字
case 'a':
if ( rcp(curpos, 0) == 'u'
&& rcp(curpos, 1) == 't'
&& rcp(curpos, 2) == 'o'
&& !(map[rcp(curpos,3)]&(DIGIT|LETTER)))){
curpos += 3;
newToken("auto", KEYWORD, AUTO);
return; }
goto id;

//尝试匹配break关键字
case 'b':
if ( rcp(curpos, 0) == 'r'
&& rcp(curpos, 1) == 'e'
&& rcp(curpos, 2) == 'a'
&& rcp(curpos, 3) == 'k'
&& !(map[rcp(curpos,4)]&(DIGIT|LETTER)))){
curpos += 4;
newToken("break", KERWORD, BREAK);
return; }
goto id;

... //尝试匹配其它可能关键字

//非关键字,是变量名
case 'h': case 'j': ... case 'z':
case 'A': case 'B': ... case 'Z':
id: {
QString stmp = QString(chcode);
while((curpos<m_len) &&
(map[rcp(curpos,0)]&(DIGIT|LETTER))){
stmp += (m_text.at(curpos));
++curpos;}
newToken(stmp, IDENTIFIER, VARNAME);
return; }
default:
QString str(chcode);
newToken(str,IDENTIFIER,VARNAME);
break;
}
}
```

图 2 单词提取主算法

对于 a 开头的字符,需往后探测 4 步,判断是否为 auto 关键字;对于 b 开头的字符,需往后探测 5 步,判断是否为 break 关键字,依次类推,探测 case、char、const、default、double 等关键字,如果不是关键字或首字符例如 h、j、A 等,则跳转到变量命名 id 处理,提取为变量名。提取出的 CToken 实例存放到单词链表,供后续脚本执行模块使用。

2.2 脚本执行模块

脚本执行模块通过分析单词链表,提取 if 控制语句段和各个函数段。根据预先定义的脚本函数名提取出该函数包括的单词子链表。当扫描到 if 关键字时,根据 {、} 的偶对匹配,提取 if 控制语句包含的脚本函数,某分支条件成立时,执行该分支内的脚本序列。之后进行类型划分,将脚本放到不同的优先级队列,在保护测控装置驱动包处理程序的不同阶段,插入执行对应优先级的脚本函数。不同优先级脚本执行时序如图 3 所示,共分 8 个队列,优先执行插件型号设置、元件

和页面投退的脚本函数。这是由于部分插槽是多选型号,可根据实际需求通过 SetBoardType 脚本设置当前投入的插件型号,而部分页面的有效性和插件型号关联,例如某页面的功能是否执行的表达式为“B08_BOARD_TYPE == NR1501”,则需根据 B08 的当前型号进行替换和使能表达式的计算,通过执行如 Enable、Disable(B02, AdcSample) 的脚本实现元件、页面投退,可刷新变量库中变量的状态,被退出的信号不输出到最终的产物文件中。

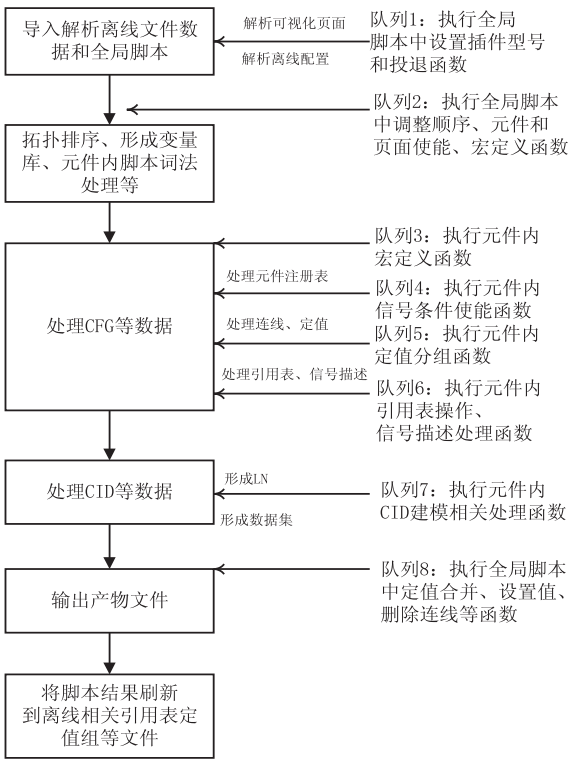


图 3 脚本执行过程

2.3 表达式计算模块

If 语句、部分脚本函数都有表达式,表达式支持算术四则运算、逻辑运算、C 库函数、系统软件平台定义的接口函数,表达式中可包括宏、定值。当条件为 1 时执行该分支或函数,否则不执行。例如处理 IEC61850 的数据集时,需判断信号的显示属性是否对 IEC61850 进程可见,此时需计算信号的显示属性表达式。表达式计算流程如图 4 所示。

在图 4 中,为避免重复计算,先从缓存 hash 表中查找表达式计算值,如果找到则返回,首次计算后,将表达式和值存入 hash 表中。对于扩展格式的表达式,需进行预处理工作。例如对于包括:的形如“B02. I1n: B02. I2n”的表达式,根据当前定值是取 1 次侧的值或取 2 次侧的值,提取其中 1 侧进行运算。对于层次变量名,例如 B02. I1n,用该定值的设置值进行替换。最后根据 C 语言的优先级采用递归下降的算法计算得出值。

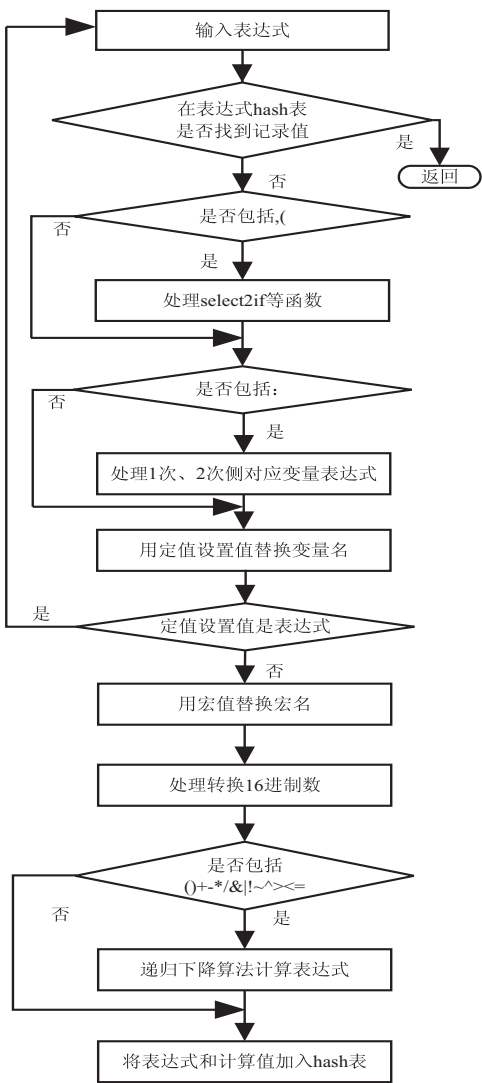


图4 表达式计算流程

3 应用实例

3.1 元件脚本应用

以过流保护 OverCurrent 元件为例,过流元件实例化成 OC1、OC2 等多个实例后,每个实例的变量都有相同的配置需求,例如部分变量配置到报告菜单、录波、定值等。可将这些相同的操作进行抽象归类,以脚本的形式形成可共享的 API 函数库,提供给模块研发人员编写。在模块化的元件内增加脚本段,用于定义单个元件的配置规则,包括引用表添加、定值分组、逻辑节点配置等功能。过流元件的脚本段定义示例如图 5 所示。

```
/变位引用表配置
AddTab(binchg_refer_table, oc_flag);
//定值分组
AddParaGroup(OC_Settings, 1, oc_mode);
//变量和逻辑节点关联
SetDAI(oc_flag, GGIO1.SPCSO1.stVal);
```

图5 元件脚本应用

在图 5 中,AddTab 是配置引用表功能的脚本函数,把变量 oc_en 添加到 binchg 引用表;AddParaGroup 是配置定值分组的脚本函数,把变量 oc_mode 添加到定值组 OC_Settings;SetDAI 是映射变量和逻辑节点数据属性短地址的函数,表示把变量 oc_en 填充到 GGIO1.SPCSO1.stVal 的 sAddr 属性。

3.2 全局脚本应用

以装置选型 MOT 全局为例,阐述全局脚本的使用过程:

(1)定义功能选型配置文件格式,装置研发人员在研发版本配置软件的功能选型编辑界面中,以编辑层次树的方式,完成选型问题设计划分,输入问题的若干候选项和默认值。可设置问题选项的强制、互斥等条件表达式,根据其他问题的当前选项值设置某个问题强制选项值或灰化互斥部分可选项。条件表达式是问题名、选项值、常量、全局变量的逻辑组合语句。

(2)使用脚本函数和规则文本接口,装置研发人员在研发版本配置软件的功能选型编辑界面中,编写和问题、选项关联的脚本。支持使用的全局脚本函数包括:设置插件型号 SetBoardType、显示/隐藏插件 ShowBoard/HideBoard、修改系统配置选项 SetSysCfg 等。

(3)用户在用户版本配置软件中,根据实际工程配置需求,选择每个问题的对应选项。每个选项的切换会触发实时脚本执行引擎,执行和选项对应的脚本,实现硬件型号切换、软硬件功能模块投入和退出、可视化页面隐藏和显示、变量属性修改等功能。

全局脚本应用如图 6 所示,当用户切换每个问题的当前选项时,用户版本配置软件将脚本中的问题名称用当前选项替换,并调用脚本执行引擎,执行相关脚本函数。

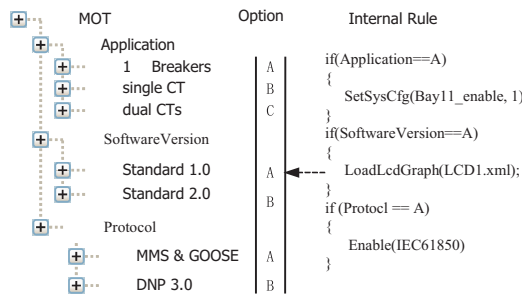


图6 全局脚本应用

4 结束语

介绍了基于元件脚本、全局脚本的电力系统嵌入式装置的可配置方案,装置研发人员可开发通用的功能配置,并提供可选功能描述和选项,用户根据实际工

(下转第 180 页)

系统能效达到最大。仿真结果证明了理论分析的正确性。

参考文献:

- [1] Mitola J I, Maguire G Q J. Cognitive radio: making software radios more personal [J]. IEEE Personal Communications, 1999, 6(4): 13-18.
- [2] Mitola J. Cognitive radio: an integrated agent architecture for software defined radio [D]. Sweden: KTH Royal Institute of Technology, 2000.
- [3] 郭彩丽, 张天魁, 曾志民, 等. 认知无线电技术的国内外发展和研究现状[J]. 现代电信科技, 2006(6): 29-34.
- [4] 王 军, 李少谦. 认知无线电: 原理、技术与发展趋势[J]. 无线电技术与信息, 2007(7): 27-31.
- [5] 沈爱国. 认知无线电技术分析[J]. 电信快报: 网络与通信, 2014(7): 12-16.
- [6] 谭学治, 姜 靖, 孙洪剑. 认知无线电的频谱感知技术研究[J]. 信息安全与通信保密, 2007(3): 61-63.
- [7] 王颖喜, 卢光跃. 基于最大最小特征值之差的频谱感知技术研究[J]. 电子与信息学报, 2010, 32(11): 2571-2575.
- [8] 赵东峰, 刘 涛, 周贤伟. 滤波器组的多滤波器联合能量频谱感知算法[J]. 电波科学学报, 2009, 24(6): 1146-1149.
- [9] Cabric D, Mishra S M, Brodersen R W. Implementation issues in spectrum sensing for cognitive radios [C]//Conference re-

cord of the thirty-eighth Asilomar conference on signals, systems and computers. [s. l.]: IEEE, 2004: 772-776.

- [10] 卞 荔, 朱 琦. 基于数据融合的协作频谱感知算法[J]. 南京邮电大学学报: 自然科学版, 2009, 29(2): 73-78.
- [11] Guo C, Peng T, Xu S, et al. Cooperative spectrum sensing with cluster-based architecture in cognitive radio networks [C]// Vehicular technology conference. [s. l.]: IEEE, 2009: 1-5.
- [12] Chen Y. Analytical performance of collaborative spectrum sensing using censored energy detection [J]. IEEE Transactions on Wireless Communications, 2010, 9(12): 3856-3865.
- [13] Peng T, Guo C, Wang W B. Energy-efficient cooperative spectrum sensing in cognitive radio networks [J]. Journal of Beijing University of Posts and Telecommunications, 2010, 33(4): 93-96.
- [14] Zhang J, Zheng F C, Gao X Q, et al. Sensing-energy efficiency tradeoff for cognitive radio networks [J]. IET Communications, 2014, 8(18): 3414-3423.
- [15] Liang Y C, Zeng Y, Peh E C Y, et al. Sensing-throughput tradeoff for cognitive radio networks [J]. IEEE Transactions on Wireless Communications, 2008, 7(4): 1326-1337.
- [16] Wang C W, Wang L C, Adachi F. Performance gains for spectrum utilization in cognitive radio networks with spectrum handoff [C]//International symposium on wireless personal multimedia communications. [s. l.]: [s. n.], 2009.

(上接第 175 页)

程需求进行功能可控选择, 通过内置脚本, 自动执行相关函数, 修改配置数据和界面显示。基于可配置、可复用的脚本, 显著减少了装置二次开发工作量, 提高了研发效率和产品的通用性、适用范围, 已经在电力系统保护测控装置中大规模适用, 取得了较好的经济效益。

参考文献:

- [1] 贺 敏, 陈宏君. 组件开发架构及在继电保护配置软件中的应用[J]. 软件工程, 2016, 19(1): 35-38.
- [2] 李 响, 刘国伟, 冯亚东, 等. 新一代控制保护系统通用硬件平台设计与应用[J]. 电力系统自动化, 2012, 36(14): 52-55.
- [3] 孙振华, 高传发, 任华锋, 等. 新型基于 MPC8309 的微机继电保护平台[J]. 计算机系统应用, 2015, 24(3): 105-109.
- [4] 陈翔宇, 王冬青, 李 刚, 等. 基于通用平台的智能变电站一体化设计和整合应用[J]. 电网技术, 2014, 38(1): 58-62.
- [5] 郭 玮, 田录林, 张永良, 等. 基于嵌入式 PLC 软核的通用保护平台设计与实现[J]. 电力系统保护与控制, 2014, 42(16): 122-126.
- [6] 陈宏君, 刘克金, 冯亚东, 等. 新一代保护测控装置配套工具软件设计与应用[J]. 电力系统自动化, 2013, 37(20): 92

-96.

- [7] 徐 兵, 沈玉利, 谢仕义. 嵌入式 Web 服务器端脚本引擎设计与实现[J]. 计算机工程与设计, 2008, 29(15): 3933-3935.
- [8] 屈景怡, 陈钟玉, 吴仁彪. 基于自定义脚本的适配参数系统的设计与实现[J]. 计算机工程与设计, 2015, 36(11): 3134-3139.
- [9] 张 强, 郭丽丽, 马振林. 基于模型自动判读的研究与实现[J]. 计算机技术与发展, 2014, 24(7): 17-20.
- [10] 何 啸, 麻志毅, 王瑞超, 等. 语义可配置的模型转换[J]. 软件学报, 2013, 24(7): 1436-1454.
- [11] IEC/TC57. Communication networks and systems for power utility automation-part 6: configuration description language for communication in electrical substation related to IEDs [S]. [s. l.]: [s. n.], 2009.
- [12] IEC/TC57. Communication networks and systems for power utility automation, part 7-4: basic communication structure-compatible logical node classes and data object classes [S]. [s. l.]: [s. n.], 2010.
- [13] MediniQVT project [EB/OL]. 2012. <http://projects.ikv.de/qvt>.
- [14] ATL flow project [EB/OL]. 2013. <http://opensource.urszeidler.de/ATLFlow>.