

基于低代码的DCS可自定义自动化设计

Customizable DCS automation design software based on low-code technology

张旭*, 姚璋

ZHANG Xu*, YAO Zhang

(中国核动力研究设计院 核反应堆系统设计技术重点实验室, 成都 610213)

摘要: 随着智能制造的发展, 自动化设计工具在大规模DCS工程设计过程中的作用日趋提升。针对辅助工具需求的快速变更与软件开发的慢速响应之间的矛盾, 基于低代码设计理念构建了DCS可自定义自动化设计软件, 让工程设计人员可以根据实际需求快速开发适用工具, 降低了对专业软件开发人员的依赖, 有效提升了工作效率。软件接口设计可匹配不同的图形组态工具与工程设计软件, 具备良好的可移植性。软件应用于福建漳州核电1、2号机组安全级DCS供货项目设计过程中, 取得了良好的效果。

关键词: 低代码; 代码生成; 图形组态

中图分类号: TP39

文献标志码: A

文章编号: 1009-0134(2023)08-0007-05

0 引言

在仪控系统的工程设计中, 计算机辅助设计(CAE)可极大地提高工程设计人员的效率。智能制造概念带来的企业数字化转型需求, 对软件设计的应用开发方式提出了新的要求。传统的工程设计通常需要基于成熟的商用设计软件进行二次开发, 需要针对项目需求进行定制化改进, 在实际应用中让工程设计人员参与到辅助设计软件的开发更加贴合实际需求, 可以避免实现过程中的信息传递偏差。灵活开发、多元化定制的理念藉由“低代码”设计的开发思路可以很好地实现^[1-5]。

“低代码”(Low-code)技术的概念诞生于2014年, 在研究机构Forrester Research提出的定义中将低代码描述为一种可通过简单配置部署实现快速开发应用、无需编写代码的技术^[6-8]。原生代码通常由专业软件开发人员根据业务需求进行开发, 开发过程中时常会由于软件设计人员对于客户领域的业务流程或设计目标的不熟悉而造成理解上的偏差, 最终影响代码的功能实现。低代码设计由工程设计人员自主实现, 有助于消除理解误差并提高简单需求开发的效率。低代码设计理念的诞生改变了常规的软件开发流程, 自此之后可以通过将软件开发的工作分为元数据模型开发和特定功能域开发两部分, 工程设计以元数据模型为载体并通过模型化及特定域表达来执行应用^[9,10]。

当前低代码开发应用场景主要包括三类: 数据管理、界面设计、表单编制。在这几类低代码平台中, 用户可以根据实际业务需要将图形化的业务逻辑进行任意的组合搭配, 实现特定的功能逻辑。通过低代码平台可以跳过代码开发的过程, 直接使用预设好的功能模块快速完成目标。本文所设计的低代码平台将低代码概念拓展到了工程设计

领域, 在仪控系统如DCS的设计过程中, 为不熟悉代码开发的工程设计人员提供了一种便捷有效的解决方案, 无需专业的软件开发人员即可进行定制化的开发。

1 总体设计

1.1 基本功能设计

基于低代码的DCS可自定义自动化设计的总体流程如图1所示, 包括平台开发和平台应用。其中, 平台开发包括了流程建模和功能模块设计, 平台应用则由图形组态、代码生成和代码编译构成。



首先对流程建模, 将自动化设计流程抽象分解为功能、步骤、关系和选项四部分, 对业务架构、数据架构和应用架构进行全面描述, 并以图形组态的方式完成功能搭建。在图形组态中, 页对应实现了某种特定功能, 图形块对应了功能页中各个步骤, 各个步骤间的逻辑联系映射为连接线, 可填充属性对应模块的参数选项。以Microsoft Visio作为前端组态界面为例, 在Visio中使用模板可以轻松直观的创建功能模块, 使用者仅需将预设的功能模块拖拽至工作区并连接端口, 即可搭建完整的自定义功能流程图。功能模块基于以下要求设计^[11]:

- 1) 高内聚: 模块具有强内相关性, 对外负责特定一项任务, 即单一责任原则;
- 2) 低耦合: 模块可根据设计变化进行合理功能划分, 具有较强复用性;

收稿日期: 2021-11-08

基金项目: 四川省杰出青年科技基金(2020JDJQ0068); 核反应堆系统设计技术重点实验室运行基金

作者简介: 张旭(1992-), 男, 吉林人, 工程师, 硕士, 研究方向为核安全级仪控系统、自动化工具软件开发。

3) 自包含: 模块是特定功能逻辑的整体, 可独立进行编译。

完成图形组态后, 将图形化组态映射翻译为可用代码。代码转换过程要将图形化组态中的功能块、连接线、逻辑顺序等要素按照特定规则进行映射。映射关系如图2所示, 功能页对应了函数框架主体, 功能块对应子函数, 连接线根据实际组态体现函数中的变量关系, 属性则是函数中固定的常量。代码生成成功后通过编译器编译为动态链接库格式, 即可通过加载至工程设计软件如Eplan中的API接口运行, 或根据其他CAE软件的特性, 生成适应性的自动化设计软件。

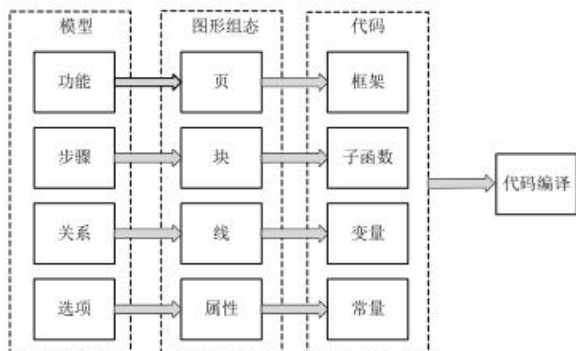


图2 业务流程逻辑

1.2 软件应用平台

基于低代码的可自定义工程软件设计平台具有以下特性: 能够以图形可视化方式构建业务对象流程图, 自定义输入要素及配置关联; 能够灵活配置用户角色及权限, 并将其应用于逻辑关系; 能够封装应用后分发给对应的用户; 能够设置个性化的工作界面^[12]。

平台架构如图3所示, 元数据模型的开发及维护在管理服务器中进行。作为PaaS (Platform as a Service, 平台即服务) 功能的推广, 管理服务器面向工程设计人员提供图形组态应用服务, 返回给使用者适配CAE软件二次开发方式的自动化设计软件。以Eplan为例, 对应生成动态链接库格式的算法库, 并可直接加载进行使用。服务器与客户端组成一个局域网, 储存服务器与维护服务器分开部署以提高响应能力, 同时通过隔离网关与信息安全服务器及互联网相连, 一方面可以使用互联网的便捷资源, 一方面提高信息安全防护能力。信息安全作为与整体功能保障的一部分, 可以确保服务器中所存储的元数据模型、算法库等关键信息可以正确地实现预定功能, 不会受到来自互联网或其他方面的恶意攻击。

2 功能设计

2.1 功能完备性分析

对DCS工程设计过程中自动化需求进行分析, 主要类别包括图纸及清单的自动制作、自动检查、以及其他图纸

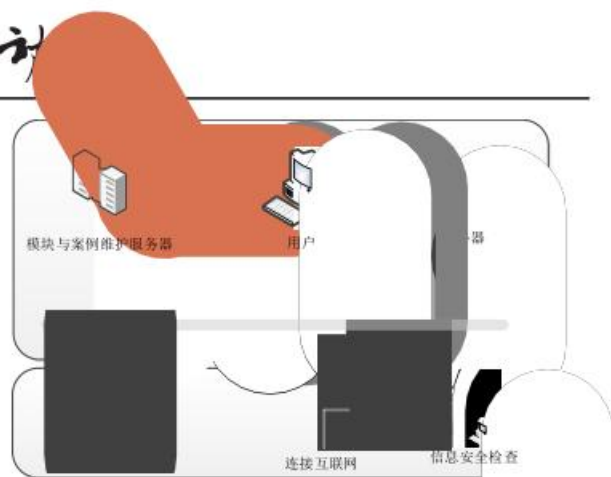


图3 应用平台架构

管理功能。实现上述功能的步骤可以分解为对图纸的直接操作和逻辑处理, 并可进一步分解至读取、筛选、比较、合并等微功能, 通过不同的微功能组合来全面覆盖工程需求^[13, 14]。

基于Visio软件在前端人机接口中设计了一系列功能模块, 如图4所列举的部分模块, 包括读取、筛选、合并、生成等。功能模块的绘制建立了可应用的标准模板, 在模块的形状清单Shapesheet中填写所需的属性, 可以自定义模块的输入输出特征。



图4 功能模块

2.2 图形解析功能设计

图形解析的内容应包括功能模块及其属性、连接线及其对应模块的端口等信息。图形解析流程包括以下步骤:

- 1) 导入图形组态, 通过Visio接口读入组态数据;
- 2) 读取所有功能模块及其属性, 包括图符类型、输入输出关系、形状数据等;
- 3) 读取所有连接线及其属性, 包括连线编号、连线名、起点信息、终点信息等。

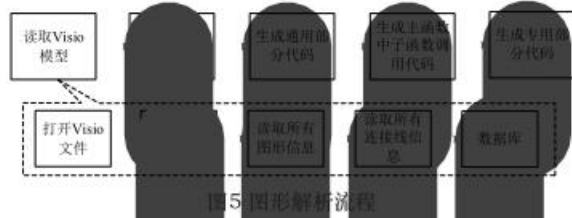


图5 图形解析流程

以C#语言读取Visio文件为例, 读取Visio组态代码如下:

```
Visio. Application app = new Visio.
Application();
Visio. Document doc = app. Documents.
Open(fname); doc = app. Documents. Open(fname);
```

```
Visio. Shapes shapes = ((Visio.Page) ((Visio.Document) doc). Pages[1]). Shapes;
```

```
Foreach (Visio. Shape shape in shapes) {
    ...}
```

读取Shapsheet中的属性（读取动态连接线的起点端）示例代码如下：

```
if (shape. get_CellExists ( "BeginX", 0) != 0) {
    start_block = shape.Cells[ "BeginX" ].
    Formula. Split( '!' ) [0]. Split( ' ( ' ) [2];
}
```

为实现用户预期的功能，首先应可靠地实现对图形描述逻辑的解析。图形解析过程中读取到的图形组态信息存放在功能模块链表中，图形组态中的每个功能模块都影响后续代码生成。为保证生成代码的逻辑正确性，调用子函数应符合逻辑先后顺序。由于在功能模块链表中存放数据的顺序与用户放置功能模块的顺序有关，而非按照逻辑上功能模块的先后顺序，但代码翻译的顺序应严格遵循逻辑顺序，采用递归算法将功能模块按逻辑顺序重新排序。递归算法对所有功能模块执行查询，循环从任一功能模块开始，沿该块的输入逻辑树逆向查询，得到以某一功能模块为输出的一串功能模块并置入链表，全部循环一次后即可得到逻辑正确的代码段。



图6 图形组态排序方法

2.3 代码生成功能设计

生成的代码结构如图7所示，为降低代码的冗余度并提升其可维护性，应包括公共代码部分、主函数代码部分和子函数代码部分。



图7 生成代码结构

公共代码头主要包括引用文件，构成基本函数库。主函数代码包括函数初始化、基本定义、代码基本结构体等，其中的待替换内容即对应了图形组态转换成的代码。其它公共代码为子函数调用函数，对涉及子函数代码进行调用。根据模块化设计的思想，子函数为最小的颗粒，不因图形组态的连接关系而改变，并可在通用算法库中获取，保证本功能中涉及子函数能够在本次代码生成中被引用。

生成专用部分代码如下：

```
Former Code = Former Code. Replace( "//内容部分待替换", Main Code);
```

```
File Stream fs = new File Stream (File Path +
    "\\ " + File Name + "." + Extension, File Mode.
    Create, File Access. Read Write);
```

```
Stream Writer sw = new Stream Writer (fs,
    Encoding.UTF8);
```

```
sw. Write Line (Former Code);
```

```
sw. Close ();
```

```
fs. Close ();
```

在代码生成过程中，为了保证函数语句间变量传递的连续性，生成主函数中组态对应代码时，应注意各函数输入输出变量的一致性，即对于前后逻辑相连的两个子函数，前一个函数的返回值变量名应与后一个函数参数的变量名一致，保证变量处理信息的完整性。同时为避免用户手动为每个输入输出变量命名的不便，考虑内部变量不影响用户功能的实现和对功能的理解，采用Visio中每根连接线自动进行编号加以区分。例如，采用英文字符与该数字的结合，可保证输入输出变量的对应关系。

2.4 接口设计

自定义自动化设计软件设计与多个软件的接口，以保证功能的完整性和用户使用的便捷性。

如图8所示，描述了低代码系统与Visio和Eplan软件的接口关系。与Visio的接口实现图形解析的功能，引用算法库即可定义进程对象和Visio文件对象，进而解析Visio对象。借助Visio软件优异可靠地绘图功能呢，为用户提供体验良好的图形组态平台。代码如下：

```
using Microsoft Office Interop Visio
```

```
Visio. Application app = new Visio.
    Application();
```

```
Visio. Document doc = app Documents.
    Open(filename);
```

与Eplan的接口实现用户预期的软件功能，Eplan加载低代码系统编译生成的动态链接库，即可执行相应的功能^[15]。



图8 低代码系统接口设计

2.5 容错设计

自定义的低代码功能设计给予用户极大的自主性，这也导致了用户使用过程中可能出现各种非预期的使用错误。针对此类问题，应确保系统能够及时发现，避免生成错误的代码以致无法执行设计功能或导致不可预估的结果。

当用户使用的功能模块并非算法库中合法的功能模块时，系统在生成代码前即检查功能模块合法性并予以告警，拒绝生成代码，保证设计功能实现过程中的可靠性和安全性。

当连接线未连接到功能模块时，将导致函数输入输出关系不完整，所生成的代码必然不能正常编译。针对该情况同样应该及时告警并提示用户修改。

3 实验验证

通过以下三个实例，分析平台的可用性和对DCS自动化设计过程的适用性。

3.1 图纸升版整合功能实例

需求示例1：工程设计图纸升版，为了明确修改部分，应遵循只升版修改页的原则，例如A版中部分图纸升版为B版时，需将A版中有变更内容的图页替换为B版图页，因此要将已签署的A版部分未修改图页与B版有内容修改的图页合并成新的B版图纸作为最终文件。

针对需求示例1，在前端界面中使用了读取图页、筛选页、筛选码、读取PDF、生成PDF、合并PDF等功能模块，通过连线搭建实现流程如下：读取B版图纸中本次升版的图页并生成PDF-读取A版已签署的PDF格式图纸-筛选A版PDF中需要进行升版的页码-合并PDF得到最终PDF格式文件。

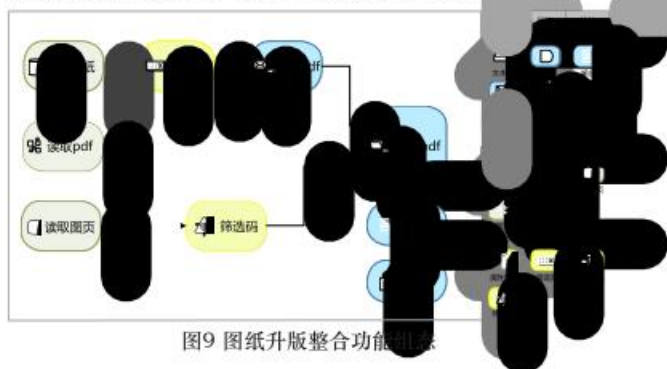


图9 图纸升版整合功能流程图

生成子函数代码示例如下：

```
public bool Execute(ActionCallingContext
o ActionCallingContext){
    Selection Set ss = new Selection
```

```
Set();
    Project pj = ss. Get Current
Project(true);
    string a21;
    a21 = Read PDF(“F:/测试工程/机柜
原理图.pdf”);
    List<Page> a2 = new List<Page>();
    a2 = Read Pages(“原理图”);
    List<Page> a3 = new List<Page>();
    a3 = Read Pages(“原理图”);
    List<Page> a27 = Select Page(a2,
“B”);
    List<int> a22 = Select Page Int(a3,
“B”);
    string a24;
    a24 = Generate PDF (a27, “D:/
EplanAddIn/GeneratePDF.pdf”);
    Combine PDF (a21,a22,a24, “D:/
EplanAddIn/GeneratePDF2.pdf”);
    return true;
}
```

3.2 清单生成功能实例

需求示例2：在DCS工程设计中，需根据图纸编制相应的设备材料清单、IO清单、端接清单等表格类文件，上述文件的制作过程工作量大且规律性强，适合通过软件实现清单的自动生成。清单生成功能的自动化辅助功能软件不需要专业软件开发人员通过复杂的逻辑实现，因而可由工程设计人员自行通过低代码平台开发。

针对需求示例2，利用读取图纸和生成表格两个功能模块即可完成搭建，对业务流程建模。在读取图纸功能模块的属性中定义图纸类型，然后在表格生成模块中填写每列的内容，如变量名、变量描述、机柜编号等，同时在逻辑属性中规定各列属性间的逻辑关系，具体可根据实际清单需求自定义实现。

A列	变量名
B列	描述
C列	机柜号
D列	设备号
E列	接口号
F列	
G列	
H列	
输出	D:/EplanAddIn/lowtable
逻辑	E的变量名=A的变量名,F的

图10 生成表格功能属性

生成子函数代码示例如下:

```
public bool Execute (ActionCallingContext
o ActionCallingContext)
{
    Selection Set ss = new Selection
Set();
    Project pj = ss. Get Current
Project(true);
    List<Function> a3 =new
List<Function>();
    a3= Read Project(“原理图”);
    Generate Table(a3,“变量名”,“
原理图”,“描述”,“机柜号”,“设备号”,“接口
号”,“,“,“,“,“,“,“,“D:/EplanAddIn/table.csv”,“E
的变量名=A的变量名;F的变量名=A的变量名;H的变量名=A
的变量名”);
    return true;
}
```

3.3 图纸检查功能实例

需求示例3: 工程设计图纸编制时, 需确保原理图设计与布置图设计的一致性, 对原理图中所有设备部件应在布置图中提供安装位置和相应接口。为防止布置图纸中遗漏部件, 传统做法采取通过人工互查、校对等步骤的方式来进行, 自动化检查可以更便捷的达到目的并提高检查效率。

针对需求示例3, 在前端界面中使用了读取图页、属性筛选、比较等功能模块, 通过连线搭建实现流程如下: 读取原理图图纸-筛选原理图中需要检查的元素-读取布置图图纸-筛选布置图中需要检查的元素-比较-输出不一致的元素内容。

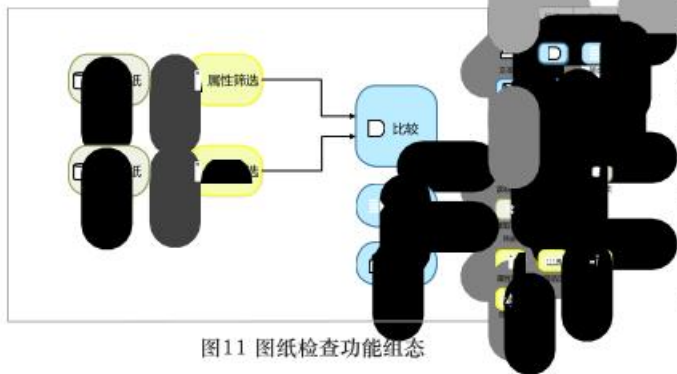


图11 图纸检查功能组态

生成子函数代码示例如下:

```
public bool Execute (ActionCallingContext
o Action Calling Context)
{
    Selection Set ss = new Selection
```

```
Set();
    Project pj = ss. Get Current
Project(true);
    List<Function> a7 =new
List<Function>();
    a7 = Read Project(“原理图”);
    List<Function> a2 = new List<Function>();
    a2 = Read Project(“布置图”);
    List<Report CG> a10 = new List< Report CG>();
    a10 = Attribute Selection(a7, “元
件名”); List<Report CG> a9 = new List< Report
CG>();
    a9 = Attribute Selection(a2, “元
件名”); List<Report CG> a11 = new List< Report
CG>();
    a11= Compare2 (a9, a10);
    Output(a11, “D:/EplanAddIn/
lowcode.csv”);
    return true;
}
```

4 结语

本文设计了一套基于低代码设计理念的DCS可自定义自动化设计软件, 可面向实际工程设计中的定制化、碎片化应用场景, 让一线工程设计人员可根据实际工程需求自主化开发工程图纸自动化批量化设计功能, 而不依赖专业的软件开发人员, 减少了软件需求到实现过程中的理解偏差, 使其更加契合实际工程需求, 同时提高了开发效率。通过三个实例列举描述了软件的使用方式并证明了功能完整性, 后续可以继续丰富微功能模块, 使其可以覆盖工程应用的各个方面。目前该软件应用于漳州核电1、2号机组安全级DCS供货项目设计过程, 取得了良好的效果。

参考文献:

- [1] Marcus Woo. The rise of No/Low code software development- no experience need[J].engineering, 2020, (9):960-964.
- [2] Waszkowski Robert. Low-code platform for automating business processes in manufacturing[J].IFAC-PapersOnLine 2019, 52(10), 376-381.
- [3] A Sahay, A Indamutsa, D Di Ruscio, A Pierantonio. Supporting the understanding and comparison of low-code development platforms[J].46th Euromicro Conference on Software Engineering and Advanced Applications (SEAA), 2020, 171-178.
- [4] 任江, 阮志成, 淡雅静. 一种面向机器人自主智能导航低代码设计与仿真的实现方法[J]. 交通科技与管理, 2023, 4(13):1-4.
- [5] 高博, 何建, 郭云飞. 石油井下作业低代码成本管控系统开发与运用研究[J]. 信息系统工程, 2023(4): 28-30.

【下转第31页】

同理,利用测试仪对传感回路断线逻辑进行测试分析,其测试结果如表2所示。测试时仅对A相施加运行电流,其为任意相的运行电流,每相施加均可。

表2 回路断线判别测试结果

A相护层电流(A)	B相护层电流(A)	C相护层电流(A)	运行电流(A)	判别结果
0	10	10	/	A相断线
10	0	10	/	B相断线
10	10	0	/	C相断线
0	0	10	/	AB相断线
0	10	0	/	AC相断线
10	0	0	/	BC相断线
0	0	0	10	三相断线

从测试结论上看,所述方案能够实现盗割及传感回路断线判别,相对于光纤振动、图像识别等算法,方案算法实现简单,在终端即可实现部署且部署简单,具有良好的工程应用价值。

4 结语

本文融合开关量、护层电流、运行电流信息,采用变化量和稳态量相结合的方法,解决了现有算法复杂和误判率高的问题。同时采用护层电流横向对比并结合护层电流与运行电流相关性的判别方法,实现电缆护层电流传感器回路单相断线、两相断线和三相断线的判别功能,不但给出告警信号,而且能够明确指示断线相别,便于运行人员及时处理断线,具有良好的工程应用价值。方案对新建电缆及原有电缆均可适用,在工程应用中针对高压电力电缆已经配置护层电流等相关传感器的情况,不需增加任何硬件成本,工程实施简单。基于方案开发的产品已在天津某电力电缆工程现场运行,工程运行证明方案具有判别准确率高、可靠性高和实施简单的特点,有效提高了电力电缆盗割预警的正判率。

参考文献:

- [1] 张金平. 配电网电缆防盗技术及应用[D]. 华南理工大学, 2015.
- [2] 黄国良. 电力设备防盗智能系统技术研究与应用[D]. 华南理工大学, 2016.
- [3] 杨磊. 地下电缆防盗及故障检测系统[D]. 西安电子科技大学, 2014.
- [4] 王航, 周文俊, 陈杰, 等. 波纹金属护套高压单芯电缆线芯护层互感的解析解[J]. 电工技术学报, 2020, 35(16): 3369-3376.
- [5] 李露露, 雍静, 曾礼强, 等. 基于系统电力扰动的交叉互联电缆绝缘整体老化在线监测[J]. 电工技术学报, 2018, 33(14): 3396-3405.
- [6] 聂永杰, 赵现平, 李盛涛. XLPE电缆状态监测与绝缘诊断研究进展[J]. 高电压技术, 2020, 46(4): 1361-1371.
- [7] 袁燕岭, 周颢, 董杰, 等. 高压电力电缆护层电流在线监测及故障诊断技术[J]. 高电压技术, 2015, 41(4): 1194-1203.
- [8] 魏新芳, 朱博, 庞兵, 等. 长距离三相电力电缆绝缘在线监测方法[J]. 中国电机工程学报, 2015, 35(8): 2079-2086.
- [9] 徐伟强, 魏云冰, 路光辉, 等. 电力电缆及隧道在线监测与移动巡检协同策略探讨[J]. 电测与仪表, 2019, 56(19): 121-125.
- [10] 路光辉, 张洪涛, 刘四军, 等. 一种高压电力电缆防盗割监测方法及装置[P]. 中国: CN109035659B, 2020-02-21.
- [11] 路光辉, 李自强, 倪传坤, 等. 一种高压电力电缆护层电流传感器断线识别方法及装置[P]. 中国: CN109375028A, 2019-02-22.
- [12] 路光辉, 雍明超, 周水斌, 等. 新一代智能变电站开关综合监测装置研制[J]. 电测与仪表, 2016, 53(16): 118-123.
- [13] 雍明超, 赵宝, 路光辉, 等. 基于最小二乘法的断路器位移监测数据的拟合与处理[J]. 高压电器, 2016, 52(11): 194-199.
- [14] 雍明超, 路光辉, 牧继清, 等. 高压开关综合监测装置的设计[J]. 高压电器, 2017, 53(9): 38-45.
- [15] 谢玮, 夏水斌, 何行, 等. 基于边缘计算的电力末端融合系统的优化[J]. 电测与仪表, 2019, 56(16): 61-66.

【上接第11页】

- [6] 任建新, 刘峰, 孙杰. 低代码开发平台在企业数字化转型中的应用[J]. 电子技术, 2023, 52(2): 192-193.
- [7] 牟松, 丁献礼, 李昭君等. 低代码快速开发平台在白云机场数字化转型中的实践[J]. 电子技术与软件工程, 2022(21): 86-89.
- [8] 徐娟华. 基于可视化设计的零代码平台分析及应用探讨[J]. 铜业工程, 2020(6): 75-79.
- [9] 蔡昭权, 卢庆武, 郑宗晖, 等. 基于元数据的快速开发平台设计与实现[J]. 计算机工程, 2009, 35(9): 60-62, 65.
- [10] 曹严明, 曹蕊. 模型驱动的低代码平台及其在金融业务系统迁移中的应用[J]. 中国金融电脑, 2021(8): 60-64.
- [11] 程春蕊, 刘万军. 高内聚低耦合软件架构的构建[J]. 计算机系统应用, 2009, 18(7): 19-22.
- [12] 文建全, 李井锐, 邹驰华. 无代码平台技术及其在HIS系统中的应用[J]. 电脑与信息技术, 2020, 28(5): 30-32.
- [13] 李瑞, 王永鹏, 李昆, 等. 基于知识和模块化的工装快速设计研究[J]. 制造业自动化, 2020, 42(5): 95-97, 136.
- [14] 董玉祥. 基于模块化柔性系统的控制、仿真与教学应用[J]. 制造业自动化, 2009, 31(4): 1-3, 23.
- [15] 张旭, 黄鹏, 彭韬, 等. 基于纯模拟仿真的核级仪控系统定期试验脚本辅助设计软件的设计与实现[J]. 现代计算机, 2019, 25(32): 59-64.