一种面向汽车电子的配置界面动态生成方法

晏 华 陈 昊 郭宣佑
(电子科技大学计算机科学与工程学院 成都 611731)

摘 要 汽车电子的开发需要根据特定硬件平台资源情况对基础软件功能进行裁剪,而汽车电子的基础软件模块具有配置需求量大、复杂度高等特点。因此,设计一种具有高可配置性、通用的配置工具原型,具有非常重要的应用推广价值。针对汽车电子基础软件的实际需求,提出一种动态生成配置界面的方法。该方法分离配置参数与配置界面,从而极大提高配置工具的可扩展和可维护性。实验结果显示该方法是可行且有效的。

关键词 汽车电子,基础软件模块,配置界面,动态生成
中图法分类号 TP37 **文献标识码** A

Method for Automotive Electronics Oriented Dynamic Configuration Interface Generation

YAN Hua CHEN Hao GUO Xuan-you
(School of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu 611731, China)

Abstract The functions of basic software modules need customizing according to the resource of specific hardware platform during automotive electronics development, while the configuring requirements of automotive basic software modules are large and complex. So, designing a high configurable, general purpose configuration tool prototype has important application significance. This paper proposed a dynamic configuration interface generating method according to the practical requirements of automotive basic software. The proposed method separates the configuration parameters and configuration interface and improves the extensibility and maintenance of configuration tool greatly. The experimental results show the proposed method is feasible and effective.

Keywords Automotive electronics, Basic software modules, Configuration interface, Dynamic generation

1 引言

目前,电子技术越来越多地应用于汽车产业,并迅速成为汽车产业的主要增长点。为此,全球汽车制造商、零部件供应商以及其他半导体、电子和软件系统公司于2003年携手建立了AUTOSAR(AUTomotive Open System ARchitecture)组织,并制定汽车开放系统架构[2],即AUTOSAR标准。

汽车电子是一种分布式环境下的嵌入式系统,有着硬件资源较少、成本低、能耗低等特点。因此,AUTOSAR标准广泛使用了配置的概念,在汽车软件的开发过程中根据特定硬件平台资源情况进行配置,以达到对软件的功能剪裁和参数的配置。此外,操作系统、通信以及驱动等基础软件,作为汽车电子的基础软件平台,有大量的配置需求。为了降低人工配置的工作量,并确保配置的正确性,研究一种具有高可配置性、通用的配置工具原型,提供图形化的配置界面完成配置过程,并能解析界面配置信息自动生成基础软件的配置源代码文件,具有非常重要的应用推广价值。

创建图形化配置界面是配置工具开发的主要任务之一。传统的用户界面开发方法是在程序中显式地根据待配置参数具体需求编写许多界面定义语句,配置参数与程序高度耦合,是一种静态的界面生成方法。一旦配置参数发生改变,就需要修改程序,使得配置工具的扩展性和可维护性很差。近年来,国内外研究者充分认识到传统方法的局限性,在自动图形用户界面生成问题上开展研究并提出一些方法,从而减少用户界面开发的时间和成本。第一类典型方法是采用用户界面描述语言(大多数是基于XML的)[5,6]在更抽象的层面上描述用户界面,而将具体的、与平台相关的用户界面绘制任务交给自动系统完成。这种方法与定义一个具体的用户界面类似,需要完整指定抽象界面元素及其属性。第二类典型方法是基于模型的用户界面设计方法[7,11],采用各种模型,例如任务模型、领域模型、表示模型、对话模型等,生成用户界面。该类方法的优点在于模型不仅包含界面形式的信息,还包含生成具体界面需要考虑的语义,其抽象性可支持在开发的其他阶段提供交互信息。然而,基于模型的方法需要一些额外工作维护应用与模型的一致性,且易出错。第三类方法是基于程序注解技术,抽象隐含地将界面定义在源程序中[8-10],可解决第二类方法中应用与模型分离的问题。但是这类方法需要应用编程语言具有扩展性,即支持附加的GUI控制指令。文献[3]就采用了Java的注解技术,将界面信息注解到类模型中,通过Java的反射机制获得注解信息,从而自动生成用户

到稿日期:2012-10-09 返修日期:2013-01-30 本文受国家"核高基"重大专项(2009ZX01038-002-003),四川省应用基础研究项目(2011JY0118)资助。

晏 华(1970—),女,博士,副教授,主要研究方向为嵌入式软件、计算智能,E-mail:huayan@uestc.edu.cn;陈 昊(1988—),男,博士生,主要研究方向为嵌入式软件;郭宣佑(1986—),男,硕士,主要研究方向为嵌入式软件开发环境。

界面。将界面信息注解到模型中需要模型构建工具处理后重新生成 Java 代码[3]，当配置界面需求改变后，会需要重新生成带注解信息的模型代码，配置工具也需要再次编译、链接和发布。因此，基于文献[3]提出的方法开发配置工具虽然能实现界面的自动生成，但在工具扩展和维护上存在灵活度不够的问题。

本文针对汽车电子基础软件平台的实际需求，研究配置参数的特点，提出一种分离配置参数与配置界面自动生成界面的方法，并将其应用到实际的汽车电子软件开发过程中。结果表明该方法可行、有效，且配置工具的使用和维护更灵活。

2 基本概念

在下一代汽车电子架构下，软件系统的各个功能部分能够被拆分、重组、迁移和重新部署。应用软件的最小部署单位是构件(Software Component，swc)，通过端口和通信连接器进行交互。应用软件的设计面向虚拟功能总线(Virtual Functional Bus，VFB)进行。VFB 上的构件被分配到各个电子控制器(Electronic Control Unit，ECU)上。ECU 上的软件分为应用层、运行时环境(Run Time Environment，RTE)和基础软件(Basic SoftWare，BSW) 3 层。运行时环境是 VFB 在 ECU 上的具体代码实现，它使用 ECU 上所提供的基础软件为应用软件的通信和运行提供服务。

对于上述的架构，AUTOSAR 方法学定义了相应的工具链[1,4]，如图 1 所示。工具之间的信息交换采用标准的 XML 文档，工具分为系统级工具以及 ECU 级工具。

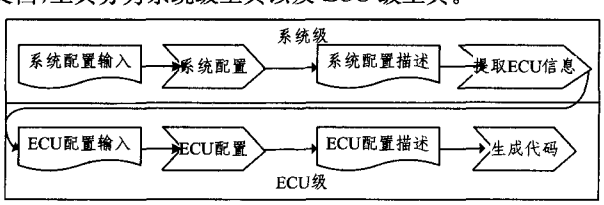

图 1 AUTOSAR 方法学定义的工具链

图 1 中的系统配置输入文件，包含软件构件描述、ECU 资源描述以及系统约束描述，确定系统具有哪些功能、硬件资源和整个系统的约束条件。AUTOSAR 提供了一系列的模板(软件构件模板、ECU 资源模板和系统模板)和标准的信息交换格式，工具供应商可据此提供相应的工具支持，从而简化系统设计的工作。

系统配置工作将初步获得的系统配置输入文件借助系统配置器生成系统配置描述文件。该文件将包含所有的系统信息，包括软件构件与 ECU 的映射关系以及通信矩阵(整车的网络结构、时序以及网络数据帧的内容)。

ECU 配置工作针对系统中每个 ECU 分别进行。首先是使用系统配置描述文件，从中提取出与各个 ECU 相关的系统配置描述信息，存放在一个单独的 XML 文件中。然后进行 ECU 配置，负责添加具体应用所必需的信息，如任务调度、必要的 BSW 模块、BSW 配置信息、给任务分配可运行实体等，配置结果存放在 ECU 配置描述文件中。最后生成具体 ECU 的可执行程序，根据 ECU 配置描述文件中的配置信息完成 ECU 基础软件的设置和软件构件的集成，最终生成 ECU 的可执行代码。

在 ECU 配置阶段，开发支持各个基础模块的图形化配置界面并自动生成配置源代码文件的配置工具是本阶段的重点。据 AUTOSAR 标准目前的版本，ECU 上的基础模块数量多达 67 个，参数数量多达上千个。更糟糕的是，AUTOSAR 标准的版本一直在更新。如果采用传统配置工具的开发方法，不仅开发效率低，而且会面临可扩展性以及可维护性差的问题。因此，本文的后续内容将针对上述问题，设计一种图形化配置界面动态生成的算法，以分离配置参数与配置界面。

3 一种动态界面自动生成算法的设计

3.1 基本思想

动态界面自动生成方法的基本思想是：分离配置界面与基础模块的配置参数信息，即将配置参数信息存储在一个配置参数信息文件中，再由配置工具中的界面生成器解析该文件，生成界面。可见，动态界面的生成主要涉及两部分：1)配置参数信息；2)界面生成器。其原理如图 2 所示。

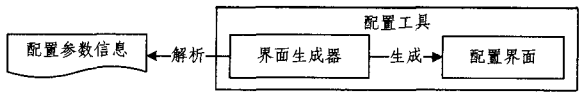

图 2 动态界面自动生成原理图

与传统的界面生成对比，该方法有一个显著的优点：当配置参数信息需要修改时，不再需要修改配置工具的源代码，也不需要重新编译链接和发布工具，而只需要按照一定格式修改配置参数信息文件，即：增加或删除模块，增加、删除或修改模块参数，甚至自定义参数文件。基于此方法开发的配置工具，不仅能支持 AUTOSAR 标准规定的参数配置，也能支持用户自定义的模块和模块参数配置，因而具有很强的可扩展性和可维护性。

3.2 配置参数信息

配置参数信息作为界面生成的依据，其内容以及内容的组织形式都需要精心设计。

AUTOSAR 定义的配置参数类型有 12 种，可分为容器类型、基本参数类型和引用类型 3 大类。容器类型包含 ModuleDef、ParamConfContainerDef 以及 ChoiceContainerDef，其共同特点是都可以包含其他类型。基本参数类型为 BooleanParamDef、IntegerParamDef、StringParamDef、EnumerationParamDef 以及 FloatParamDef 类型，其共同特点是都包含一个基本数据类型的值。余下的 ReferenceParamDef、InstanceReferenceParamDef、ForeignReferenceParamDef、SymbolicNameReferenceParamDef 以及 ChoiceReferenceParamDef 则属于引用类型，其共同特点是都包含一个或多个对其他类型的引用。配置参数除了类型，还有属性字段，例如：名称、最大数量、边界值、默认值等。配置参数以树状结构进行组织，根节点为 ECU 包，子孙节点是各个 ECU 基础模块，各模块下再包含容器、基本配置项等参数，层层嵌套。采用树状的结构适合模型化的开发模式。

3.3 界面生成器

界面生成器的主要任务是解析参数定义文件，并根据输入动态生成界面。然而，AUTOSAR 的参数定义文件只规定了各模块的参数配置项，在实际的应用中还存在各参数配置信息的存储问题，以及多次配置信息的存储问题。因此，本文定义两种文件作为界面生成器的输入，一种是 AUTOSAR 标准定义的参数信息文件，称为模型定义文件，另一种是存储参数配置信息的文件，称为模型描述文件。模型描述文件的结构由模型定义文件决定，而模型描述文件中参数值用于初始

化和保存配置界面的值。

界面生成器获得了参数定义后是如何动态生成界面的呢？为此，在界面生成器的构成中，设计了一个界面控件池以及界面管理器。界面控件池产生并管理一系列组成配置界面所需要的基本控件，而界面管理器则根据本文描述的界面生成算法，按照模型定义信息组装界面控件池提供的控件，进而完成配置界面的生成。界面生成器的架构如图 3 所示。

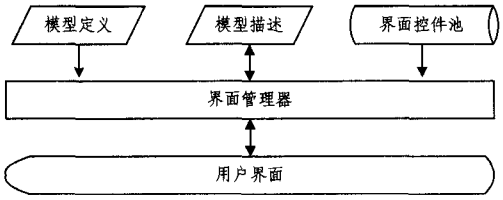

图 3　界面生成器架构图

那么，界面管理器是如何根据模型定义信息决定使用哪种控件的呢？据前文所述，AUTOSAR 模型定义文件中有 12 种参数类型，界面控制池事先定义好各种参数类型所对应的基础控件，界面管理器在生成界面过程中动态根据参数类型决定选用的控件。

界面控制池中具体设计了 4 种基本控件：editor（编辑器）、page（编辑器分页）、sectionPart（配置项分类容器）和 composite（配置页面 ui 小组件）。在这 4 类界面控件中，editor 是配置界面的核心，具有布局配置界面的功能。同时，它是配置界面中的最大分类，包含一个或多个 page 页面，对应模型定义中的 ModuleDef 或 ParamConfContainerDef。page 页面主要包含一个或多个 sectionPart，并对 sectionPart 进行布局。它包含于 editor 中，是 editor 不同页面的展示，对应于 editor 中模型定义的子模型。sectionPart 则对配置界面中的配置参数信息进行分类，以方便用户进行配置操作。具体来讲，sectionPart 分为 5 类，对应的配置参数信息分类如表 1 所列。

表 1　sectionPart 分类

类型	分类信息
TableSectionPart	AUTOSAR 标准中需要进行多次配置的配置类型
ParamSectionPart	AUTOSAR 标准中的基本参数配置类型
ContainerSectionPart	AUTOSAR 标准中的子容器配置类型
ReferenceSectionPart	AUTOSAR 标准中的引用配置类型
ChoicesSectionPart	AUTOSAR 标准中的选择型配置类型

composite 是配置界面的最小组成单位，对应模型定义中的所有基本参数类型和引用类型。对于模型定义中的每一种类型，界面控制池中都为其实现了对应的 composite 组件来进行界面显示。具体的 composite 类如表 2 所列。

表 2　composite 分类

composite 类型	模型定义
BooleanValueComposite	BooleanParamDef
EnumerationValueComposite	EnumerationParamDef
FloatValueComposite	FloatParamDef
StringValueComposite	StringParamDef
IntegerValueComposite	IntegerParamDef
ChoiceReferenceValueComposite	ChoiceReferenceParamDef
ForeignReferenceValueComposite	ForeignReferenceParamDef
InstanceReferenceValueComposite	InstanceReferenceParamDef
LinkerSymbolValueComposite	LinkerSymbolDef
ReferenceValueComposite	ReferenceParamDef
SubContainerComposite	ParamConfContainerDef
SymbolicNameReferenceValueComposite	SymbolicNameReferenceParamDef

表 2 列出的 composite 组件由不同的 SWT/JFACE 组件组合而成，一些是固定的组合，一些需要根据实际参数定义组合不同的 SWT/JFACE 组件。例如：如图 4 所示，IntegerValueComposite由 4 个组件组成，分别是选择按钮、配置名显示标签、配置信息文本框和进制转换按钮。IntegerValueComposite 对这 4 个基本控件进行布局并封装成一个 composite，进而组成一种最小 ui 组件。

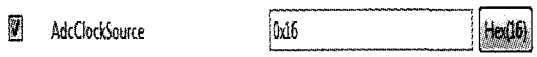

图 4　IntegerValueComposite

表 2 列出的 12 类 composite 组件中，前 5 种是基本 UI 组件，它们的 SWT/JFACE 组件组合相对固定，其组合如表 3 所列。

表 3　基本 composite 组件的 SWT/JFACE 组合

composite 类型	SWT/JFACE
BooleanValueComposite	Label，Combo，Button
EnumerationValueComposite	
FloatValueComposite	Label，text
StringValueComposite	
IntegerValueComposite	Checkbox，label，text，button

界面管理器生成的配置界面由 4 层界面控件组成，且呈树状结构显示。为了将界面控件有效组织成配置界面，界面管理器采用深度优先的方法来创建配置工具界面。首先，创建 editor 页面，editor 是配置界面的最大框架，接着再为 editor 创建每一个 page 子页面。在创建 page 子页面的过程中，递归创建 page 页面所包含的 sectionPart 和 composite。这种递归过程可以有效区分界面控件间的层次关系，并降低界面控件的耦合关系。也就是说，上层界面以底层界面的实现为基础，而底层界面则为上层界面提供服务，并屏蔽更下层的详细细节。具体的界面生成流程如图 5 所示。

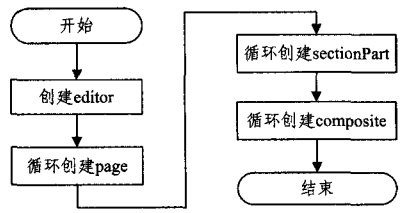

图 5　界面管理器界面生成流程

结合模型定义以及模型描述作为输入，以当前配置界面生成配置子界面为例，例如为 editor 页面生成 page 子页面，本文设计的界面生成算法 DIG（Dynamic uI Generation aglorithmn）的伪代码如下所示。

```
Algorithmn DIG
输入：ui　输出：Configuring UI
    map←∅
    get modelList of ui
    foreach model m in modelList
      get modeldef of m
      if modeldef not in map then
        add modeldef and m to map
      end if
    end foreach
    get modeldefList of ui
    foreach modeldef def in modeldefList
      if def in map then
        get model of def from map
      else
        create model of def
```

```
    end if
    create ui according to the model and def
    end foreach
```

其中 modelList 是当前界面的对应模型描述所包含的子模型描述列表，而 modeldefList 则是当前界面对应模型定义所包含的子模型定义列表。

算法首先遍历当前界面模型包含的子模型，并将该子模型及其模型定义存入哈希表中；接着再遍历当前界面模型定义包含的子模型定义，并通过哈希表检查该子模型定义是否有其对应的模型，如果没有，则为该子模型定义生成一个空的模型；最后为每一个子模型定义与其模型经由界面绘制生成界面。

4 实验结果

本节对上述的界面动态生成算法进行测试，一是验证算法的可行性和可扩展性，二是研究算法性能。测试环境为：Pentium(R) Dual-Core CPU 2.60GHz，1.99GB 内存，Windows XP professional OS。由于无法获得文献[3]配置工具代码，其实验结果以文献描述为准。此外，本文测试环境与文献[3]测试环境基本类似，但文献[3]测试处理器为 Intel(R) Core(TM)2 Duo CPU 2.66GHz，快于本文的处理器速度。

4.1 可行性与可扩展性验证

首先以 AUTOSAR 规范提供的 AUTOSAR_EcucParamDef.arxml文件为算法的模型定义文件，选取参数定义类型有代表性的 OS 模块，测试生成的界面是否符合预期。图 6 展示了算法自动生成的 OS 模块配置界面。该配置界面分为左右两个区域，左边由 OsTask、OsAlarm、OsAppMode 等标签页组成，每个标签页对应 OS 模块包含的一个容器，标签页的界面内容由容器中的参数类型决定。例如，OsTask 页包含 0…*个任务，则该页面包含一个列表。图 6 的右边是左边的容器所包含参数的配置界面。例如，图 6 左边列表中的任务 OsTask_Init 的具体配置参数包含：可激活的最大任务数、该任务的优先级、该任务是否可调度、该任务是否自动启动等等，对应的界面类型包含简单文本框、下拉列表、检查框等等。对于引用类型的参数，例如 OsTaskEventRef，算法的处理方式为其重新生成一个配置界面，如图 7 所示。

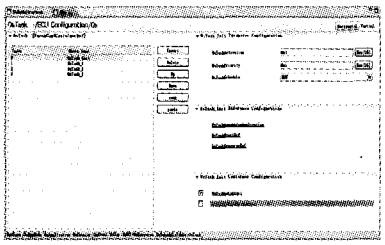

图 6 生成的 OS 模块配置界面

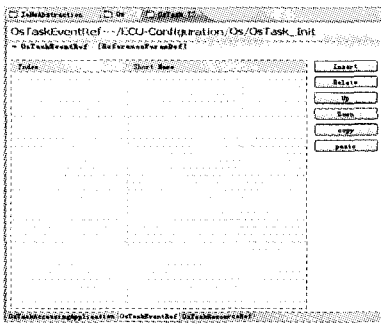

图 7 引用参数配置界面

为了验证算法的可扩展性，修改 AUTOSAR 规范提供的模型定义文件内容：1)删除大量标准模块定义；2)增加 test 模块定义，以及模块所包含的容器和基本参数。以修改后的文件作为配置工具的输入，图 8 展示配置工具在配置工程中添加 test 模块，展开树状结构显示该模块自定义的配置参数。图 9 为自动生成的 test 模块配置界面，证明了工具的可扩展性。

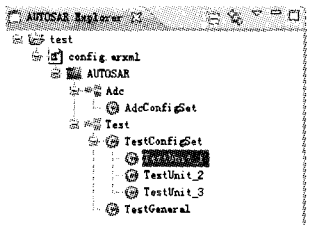

图 8 test 模块的参数树状结构

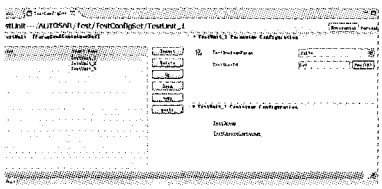

图 9 test 模块配置参数界面

4.2 算法性能分析

在算法性能分析实验中，自构造模型定义文件，通过变化参数的类型以及参数的个数，测试算法实现不同 composite ui 组件的执行时间。测试的参数类型只包含 5 种基本参数类型，即布尔型(boolean)、整型(integer)、浮点型(float)、字符型(string)以及枚举型(enumeration)。对每种参数类型，设计测试的个数分别为 100、200、300、400 和 500，测试 5 次，每次得到总的消耗时间，对 5 次总的消耗时间求和、求平均，可得到该类型生成界面平均总消耗时间。平均总消耗时间除以测试参数个数，可得到生成一个该参数类型界面所需的平均消耗时间。各基本类型参数平均总消耗时间的测试结果如图 10 所示，而生成基本参数的平均消耗时间见表 4。

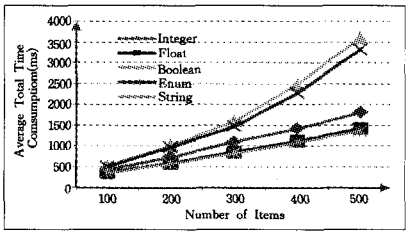

图 10 平均总消耗时间(ms)

表 4 单个参数平均消耗时间(ms)

参数类型\参数个数	100	200	300	400	500
整型	4.408	3.719	3.656	3.553	3.612
浮点型	3.592	2.97	2.853	2.812	2.844
布尔型	5.312	5.095	5.343	6.203	7.206
枚举型	5.034	4.876	4.936	5.710	6.631
文本型	3.502	2.891	2.750	2.733	2.730

图 10 显示各参数类型平均总消耗时间随参数项数增加而线性增加，其中字符串与浮点型参数消耗时间最小且几乎相同，而布尔型与枚举型参数消耗时间最大且几乎相同。测试结果符合我们的预期，即时间的多少与基本类型所使用的

(下转第 209 页)

台中投入应用，目前运行效果良好，表明了本文提出的方法是可行的和合理的，且具有较大的发展潜力。下一步的工作将对学习序列在自适应学习导航中的应用、个性化的学习模型等方面进行研究，并进一步提高原型系统的自适应能力。

参 考 文 献

[1] Iglesias A, Martínez P, Aler R, et al. Learning teaching strategies in an Adaptive and Intelligent Educational System through Reinforcement Learning[J]. Applied Intelligence, 2009, 31(1): 89-106

[2] 许高攀，曾文华，黄翠兰. 智能教学系统研究综述[J]. 计算机应用研究，2009，26(11)：4019-4022

[3] Cabada R Z, Barrón Estrada M L, Reyes García C A. EDUCA: A web 2.0 authoring tool for developing adaptive and intelligent tutoring systems using a Kohonen network[J]. Expert Systems with Applications, 2011, 38(8): 9522-9529

[4] Arnau D, Arevalillo-Herráez M, Puig L, et al. Fundamentals of the design and the operation of an intelligent tutoring system for the learning of the arithmetical and algebraic way of solving word problems[J]. Computers & Education, 2013, 63: 119-130

[5] Steichen B, Ashman H, Wade V. A comparative survey of Personalised Information Retrieval and Adaptive Hypermedia techniques[J]. Information Processing & Management, 2012, 48(4): 698-724

[6] Andre E, Finkler W, Graf W, et al. Intelligent multimedia presentations[M]// Maybury M T, ed. The automatic synthesis of multimodal presentations. Cambridge: MIT Press, 1993

[7] Sleeman A. A system which allows student to explore algorithms[C]// Proceedings of international joint conference on artificial intelligence. 1977: 780-786

[8] Murray R C, Vanlehn K, Mostow J. A decision-theoretic approach for selecting tutorial discourse actions[C]// Proc of the NAACL workshop on adaptation in dialogue systems. Pittsburgh, PA, USA, 2001: 41-48

[9] 刘箴. 智能体的一种个性模型[J]. 计算机科学，2008(12)：203-206

[10] Schiefele U. The influence of topic interest, prior knowledge, and cognitive capabilities on text comprehension[M]// Pieters J M, Breuer K, Simoms P R J, eds. Learning Environments. Springer-verlag Berlin Heidelberg, 1990: 323-338

(上接第 175 页)

SWT/JFACE 组件类型和数量相关。表 4 显示一个基本参数类型的生成时间在 5ms 左右。当前版本的 AUTOSAR 基础软件模块参数采用容器嵌套方式组织参数，一个当前配置界面中，最大不超过 20 项参数，最大生成时间不超过 100ms，非常快。因此本文提出的生成方法完全能满足汽车电子配置工具的需求。

文献[3]基本类型的 SWT 组件设计与本文设计略有不同，虽然大部分基本类型的平均总消耗时间与本文的测试结果在一个数量级上，但是 string 类型的消耗时间与本文的结果相差很大。例如：500 个 string 类型的参数界面生成需要消耗的时间在 200s 左右，而本文的消耗时间在 1.3s 左右。根据文献[3]表 2 的定义，string 类型参数采用与 Integer、Float 一样的 SWT 组件，但运行结果相差极大。

结束语 本文根据汽车电子基础软件模块配置的实际需求，提出一种动态生成界面方法，即基于 AUTOSAR ECU 参数定义模型文件，根据当前配置界面对应的参数列表及参数类型，动态生成配置界面。该方法分离了界面与参数，使得配置工具的可扩展性和可维护性很高。可行性实验选取参数定义类型有代表性的 OS 模块生成界面，其生成的界面美观、友好。性能测试实验显示生成一个基本参数类型非常快，完全满足汽车电子配置工具的需求。基于上述方法开发的配置工具不但可以支持 AUTOSAR 现有几个版本的模型文件，还可支持 AUTOSAR 标准升级版本的模型文件，而无需对配置工具作修改。

参 考 文 献

[1] AUTOSAR Partnership. AUTOSAR Methodology V1.2.2 R3.1 [S]. http://www.autosar.org/, 2009

[2] AUTOSAR Partnership. AUTOSAR ECU Configuration Parameter Definition V2.2.0 R3.1[S]. http://www.autosar.org/, 2009

[3] Li Nan, Li Hong, Zhong Xiao-feng, et al. AUTOSAR Based Automatic GUI Generation[C]// Proceedings of the 13th IEEE International Symposium on Object/component/service-oriented Real-time distributed computing. Carmona, Spain, 2010: 156-162

[4] Voget S. AUTOSAR and the Automotive Tool Chain[C]// Proceedings of the Conference on Design, Automation and Test in Europe. 2010: 259-262

[5] Puerta A, Eisenstein J. XIML: A Common Representation for Interaction Data[C]// Proceedings of IUI2002. ACM Press, 2002: 214-215

[6] Schneider K A, Cordy J R. AUI: A Programming Language for Developing Plastic Interactive Software[C]// Proceedings of the 35th Annual Hawaii International Conference on System Sciences. IEEE Press, 2002: 281-290

[7] Baron M, Girard P. SUIDT: A task model based GUI-builder[C]// Proceedings of TAMODIA 2002. Printing House, 2002: 64-71

[8] Jelinek J, Slaik P. GUI Generation from Annotated Source Code [C]// Proceedings of TAMODIA 2004. Prague, Czech Republic, 2004: 129-136

[9] Monteiro M, Oliveira P, Goncalves R. A source code based model to generate GUI: GUI generation based on source code with declarative language extensions[C]// Proceedings of the 3rd International Conference on Software and Data Technologies, INSTICC Press. 2008: 21-28

[10] Monteiro M, Oliveira P, Goncalves R. A proposal to delegate GUI implementation using a source code based model[C]// Proceedings of the 19th International Conference on Software Engineering and Knowledge Engineering. Boston, MA, Knowledge Systems Institute Graduate School, 2007: 29-32

[11] 张宁，刘正捷. 自助服务终端界面交互设计研究[J]. 计算机科学，2012，39(6)：16-20