DOI:10.19651/j.cnki.emt.2417147

基于 ARINC661 的 DF 文件状态管理方法研究

梁杭煜 赵科东 孙永荣 高玥明
(南京航空航天大学自动化学院 南京 211106)

摘 要: 在 ARINC661 座舱显示系统设计阶段,针对 DF 文件状态管理面临扩展性受限、存储效率低下等问题,提出了一种 DF 文件状态管理方法。首先,通过细化窗体部件属性结构与响应功能,构建了通用型窗体部件模型,便于窗体部件状态的统一管理;在此基础上,设计了分层动态存储结构,实现了目标状态的优化存储与快速定位;此外,针对 DF 文件状态的备份,研究了轻量化日志生成与解析技术,便于 DF 文件的版本修复。最后,利用几个典型的座舱显示画面对关键技术进行仿真测试,结果表明,该方法在设计阶段能快速准确地进行 DF 文件状态管理,具有良好的可扩展性和可靠性,且生成的状态日志相较于采用传统格式的日志,占用的内存空间减少了 17.5%以上。

关键词: 座舱显示系统;DF 文件;窗体部件模型;优化存储;版本修复
中图分类号: V241;TN27 **文献标识码**: A **国家标准学科分类代码**: 590.30

Research of definition file state management method based on ARINC661

Liang Hangyu Zhao Kedong Sun Yongrong Gao Yueming
(College of Automation Engineering, Nanjing University of Aeronautics and Astronautics, Nanjing 211106, China)

Abstract: In the design phase of ARINC661 cockpit display system, a DF file state management method is proposed to address the problems of limited scalability and low storage efficiency faced by DF file state management. Firstly, by refining the attribute structure and response function of form components, a generalized form component model is constructed, which facilitates the unified management of form component states; on this basis, a hierarchical dynamic storage structure is designed, which realizes the optimized storage and fast positioning of target states; in addition, for the backup of the DF file states, lightweight log generation and parsing techniques are investigated, which facilitates the version restoration of DF files. Finally, several typical cockpit display screens are utilized to simulate and test the key technologies, and the results show that the method can quickly and accurately perform DF file state management at the design stage with good scalability and reliability, and the generated state logs take up more than 17.5% less memory space compared with the logs using the traditional format.

Keywords: cockpit display systems;DF;widget model;optimized storage;version restoration

0 引 言

在航空电子系统领域,综合化和模块化的发展趋势日益显著,座舱显示系统(cockpit display system,CDS)也逐渐向高度集成化发展[1-2]。为了规范座舱显示系统的开发和运行,美国航空无线电协会于 2001 年提出了 ARINC661 规范,通过描述显示画面的定义文件(definition file,DF),将座舱画面显示与运行逻辑处理进行解耦,并明确定义了 CDS 与用户应用系统(user application,UA)之间的标准化通信接口[3-5]。该规范优化了 CDS 的开发流程,实现了开发周期的缩短和成本的降低,显著提升了开发效率[6-8]。

基于 ARINC661 规范的座舱显示系统设计分为 3 个阶段:开发阶段、设计阶段以及运行阶段[9]。其中,设计阶段对座舱显示画面的定义是画面正确显示和运行的基础,最终生成的 DF 文件包含了整个画面的具体描述[10-11]。

随着规范的不断普及和应用,市场上涌现出几款支持 ARINC661 规范的商业软件,如 GL Studio ARINC661、VAPS XT A661、SCADE ARINC661 等,这些软件为座舱显示系统开发提供了完整的解决方案[12]。然而,这些软件面临开发、维护涉及的安全保密以及兼容性等一系列问题,国内相关单位在适配时往往需要较长的周期。与此同时,国内各单位正在基于 ARINC661 规范的座舱显示系统开发软件国产化研究方向上努力前行。唐乐等[13]研发了一款支持部件设计与画面预览的 DF 文件设计与验证平台;吴家秀等[14]设计了一套基于 B/S 架构的 DF 文件设计与

收稿日期:2024-10-18

UA 测试系统;薛霈[15]实现了一套用于设计 Widget 库的 GUITool 图形界面开发工具。然而,这些设计平台都集中于最终 DF 文件的生成,相关论文中并没有涉及画面设计过程的监控与管理,缺乏 DF 文件状态管理方面的研究。

在座舱显示画面设计阶段,为了实现 DF 文件状态的高效管理,主要存在扩展性差、存储效率低等问题。扩展性方面,软件应具备"高内聚,低耦合"的系统特性[16],为了满足开发阶段所设定的预期画面要求,需要对大量不同类型的窗体部件进行部署[17-18],且随着座舱显示样式的丰富,部件类型也会不断扩充,扩展性问题亟需解决。存储效率方面,在实际开发中,DF 文件的内容复杂且繁多[19],对 DF 文件状态进行全信息存储将会占用大量的内存空间,导致资源浪费,且难以快速还原历史状态;此外,目前 DF 文件的设计过程缺乏高效的数据备份机制,无法保证紧急故障下的文件安全[20-21],在确保备份数据具备完整性的前提下,还应最大程度降低备份数据读取的复杂度[22]。因此,需要对座舱显示系统过程中开发人员的操作命令进行实时监控,使得开发软件具备高效的 DF 文件状态管理以及版本维护能力,降低开发时间成本,并保证 DF 文件的完整性和可靠性。

本文针对 ARINC661 规范下座舱显示系统设计阶段的 DF 文件状态监控与管理需求,提出一种 DF 文件的状态管理方法。首先,通过构建通用型窗体部件模型对 DF 文件内窗体部件进行统一管理,解决窗体部件类型扩充情况下管理扩展性受限的问题。其次,通过设计分层动态存储结构对其状态进行高效存储,提高目标状态的存储与定位效率。此外,采用基于轻量化日志生成与解析技术将历史状态数据进行日志化备份管理,解决软件发生故障后 DF 文件数据丢失的问题,最后,搭建 DF 文件状态管理平台,结合实际机载显示画面对所研究的关键技术进行验证。

1 DF 文件状态管理平台框架设计

在座舱显示画面的设计过程中,开发人员需要根据飞行员操作程序(pilot operation procedure,POP)文件对图层以及窗体部件进行逻辑与参数的定义,最终生成支持 CDS 与 UA 加载的 DF 文件[23-24]。然而,在设计过程中往往会出现属性修改的偏差导致窗体部件的显示错误,针对错误设计的定位及校正,需要对 DF 文件设计过程中的状态进行跟踪与管理,提高开发效率。

为了对 DF 文件状态进行高效管理,本文设计了如图 1 所示的 DF 文件状态管理平台,将窗体部件状态进行统一管理,并对驱动窗体部件状态变更的更新命令进行高效存储与追踪,实现 DF 文件状态的实时管理,包含以下几个模块:

1) 窗体部件设计模块:构建通用型窗体部件模型,实现各类窗体部件属性以及功能的统一管理,并对外部更新命令进行处理,实时生成窗体部件状态表,与日志内容相关联;

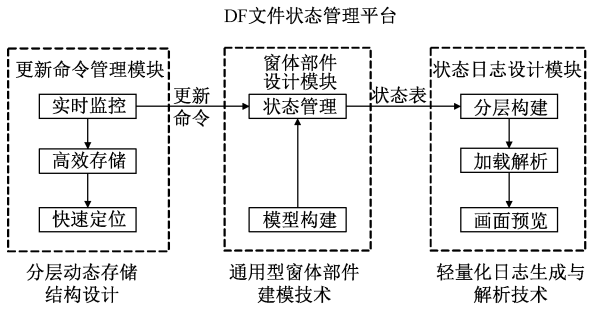


图 1 DF 文件状态管理平台框架

Fig. 1 DF state management platform framework

2) 更新命令管理模块:实时监控外部更新命令,并设计高效存储结构对其进行存储。在此基础上,针对 DF 文件的历史状态回溯,通过查询存储结构实现状态的快速定位,并作用于对应的窗体部件上;

3) 状态日志设计模块:分层构建日志文件对更新命令进行追踪,并通过建立 DF 文件版本号与窗体部件状态表相关联,实现日志文件的轻量化构建。基于生成的日志文件,通过加载解析其内容,实现存在故障 DF 文件的数据恢复,并生成恢复内容的预览画面供用户选择。

2 DF 文件状态管理关键技术

2.1 通用型窗体部件建模技术

在座舱显示画面设计阶段,窗体部件作为座舱显示画面中最小的基本单元[25-26],对其进行正确设计以及统一管理至关重要。本文设计了如图 2 所示的通用型窗体部件模型,实现窗体部件的静态属性的统一管理和动态响应的灵活适配,使得 DF 文件状态管理具备良好的可扩展性。

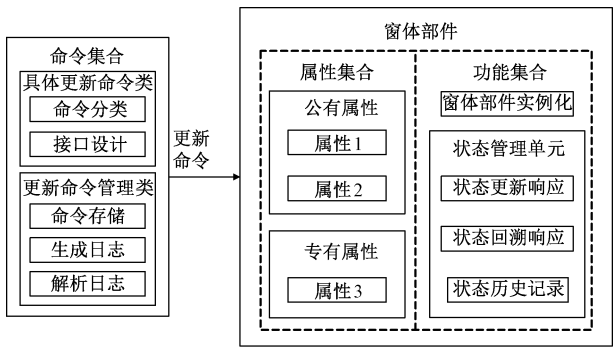


图 2 通用型窗体部件模型

Fig. 2 Generic widget model

1) 命令集合:建立具体更新命令 ChangeCommand 类,对更新命令进行分类,并设计相应的功能接口,实现具体的执行以及回溯方法,与窗体部件内部功能接口相关联;建立更新命令管理 CommandManager 类对具体更新命令对象进行存储管理,并将更新命令转换为 XML 形式日志文件。

2) 属性集合:描述了窗体部件本身所具有的外观样式特征。其中,公有属性是所有窗体部件所具有的公共属性,

在创建时进行初始化，并实行统一管理；专有属性则是不同类型窗体部件所具有的特殊属性，通过修改其属性值使得窗体部件表现出特有的状态。

3）功能集合：设计针对窗体部件属性修改以及状态回溯的响应接口，实现更新指令的快速响应。其中，针对 DF 文件中所包含的窗体部件，通过窗体部件实例化，构建其具有特定属性以及功能的实例化对象；针对捕获到的外部更新命令，通过设计状态更新接口，响应更新命令并实现窗体部件属性的更新；针对更新命令的完整性和可恢复性，通过设计状态存储接口，对历史状态进行结构化存储记录；根据状态的历史存储结构，通过设计状态回溯接口，对历史窗体部件状态进行快速定位，并将该命令重新作用于窗体部件本身。

2.2　分层动态存储结构设计

1）结构设计

座舱显示画面设计过程中存在大量的窗体部件状态更新命令，若对每个时刻的目标窗体部件及其对应的状态都进行存储，不仅需要消耗大量的内存，且查询速度较慢。为了降低算法运行过程中的存储资源占用，以及对历史状态进行快速定位，本文设计了如图 3 所示的分层动态存储结构，将窗体部件、属性、状态分别构建高效的存储单元，并根据三者的逻辑关系实现一体化的存储结构。

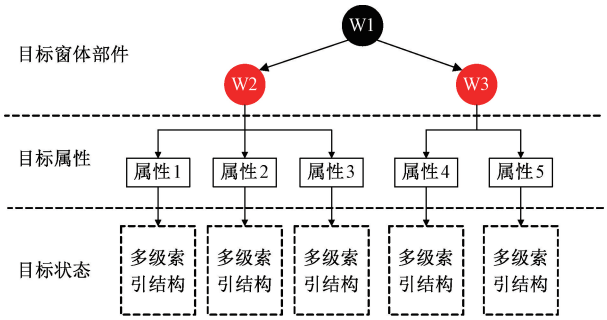


图 3　分层动态存储结构

Fig. 3　Hierarchical dynamic storage structure

其结构分成目标窗体部件、目标属性和目标状态三层，每层结构对目标对象均进行了优化存储，在减少内存占用的同时，也提高了最终目标状态定位的效率，具体如下：

（1）目标窗体部件层

通过对窗体部件添加当前最大命令序号标识，并构建红黑树将窗体部件根据标识降序存储。在该层中，当前更新状态的窗体部件需动态更新其节点位置，保证每个窗体部件在树型结构中唯一存在，且始终置于最左侧节点，从而避免重复存储同一窗体部件对象，提高存储效率。

（2）目标属性层

由于单个窗体部件的属性数量有 5～20 余个不等，在该层中，采用属性状态键值对，对当前更新命令按属性进行分类存储，同样在当前窗体部件对应属性中，添加属性集合中的最大命令序号标识，便于状态筛选后的快速定位。

（3）目标状态层

该层采用了如图 4 所示的多级状态索引结构。在每一级中，每个节点存储目标状态信息，状态节点之间以链表的形式链接，并由状态头节点进行管理。其中，同一节点出现在上一级的概率为 p，且高层级的状态节点指向低层级的状态节点，因此同一节点的高度 h 的概率分布如下：

$$P(h)=p^{h-1}(1-p) \tag{1}$$

为了减少每一层级中的状态节点数，其中 p 取 $\frac{1}{4}$。

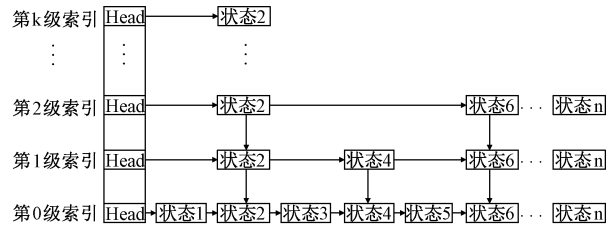


图 4　多级状态索引结构

Fig. 4　Multi-level state index structure

针对目标状态的定位，对满足当前定位序号的属性链表进行逐层检索，从最高层级依次向下查询，若大于当前节点序号且小于同层级下一节点序号，则向下一层级的同节点开始查询，以此类推，平均时间复杂度可缩减至对数级（$O(\log n)$），较采用逐个状态依次响应的方式（时间复杂度为 $O(n)$），响应速度有明显提升。

通过上述结构设计，可以实现对所属同一窗体部件的状态更新命令统一管理，并按窗体部件内部属性分类有序存储，从而提高状态存储效率，便于历史状态的快速定位。

2）性能分析

为了分析本文所设计分层动态存储结构的性能，将其与部件全信息存储以及部件状态同步存储方式进行性能比较。假设共存在 m_1 个窗体部件，m_2 个属性；令单个窗体部件对象内存占用量为 K_1，其引用内存占用量为 K_2，单个状态内存占用量为 K_3，且 $K_1 >> 2K_2$，$K_2 \approx K_3$；并对每个属性状态更改 N 次，理论上，各存储方式的性能如表 1 所示。

表 1　性能对比

Table 1　Performance comparison

复杂度	部件全信息存储	部件状态同步存储	分层动态存储
空间复杂度	m_2K_1N	$m_2(K_2+K_3)N$	$\frac{4}{3}m_2K_3N$
时间复杂度	$O(\frac{m_2^2}{m_1}N)$	$O(m_2N)$	$O(m_2\log_4 N)$

（1）部件全信息存储

针对每个更新命令，均需要存储其处于当前状态包含全信息的目标窗体部件对象。其状态存储所需的内存大小

为 m_2K_1N;由于其存储的是整个窗体部件克隆对象,需要额外利用反射机制对所有属性进行覆盖,因此,状态回放的平均时间复杂度为 $O(\frac{m_2^2}{m_1}N)$。

(2) 部件状态同步存储

针对每个更新命令,需要存储目标属性名称及其状态值,并同步存储对应的目标窗体部件本身。其状态存储所需的内存大小为 $m_2(K_2+K_3)N$;状态回放的平均时间复杂度为 $O(m_2N)$。

(3) 分层动态存储

针对每个更新命令,仅需存储目标属性名称及其状态值,且每个更新的目标窗体部件仅存储一次,并支持快速查找所需时刻的历史状态。其中,多级索引结构所存储的状态节点总数为 $N+\frac{N}{4}+\frac{N}{16}+\cdots+16+4+1=\frac{4}{3}N-\frac{1}{3}$,状态存储所需的总内存大小为 $m_1K_2+(\frac{4}{3}N-\frac{1}{3})m_2K_3$,当 N 较大时,m_1K_2 以及 $-\frac{1}{3}m_2K_2$ 可忽略不计,总内存大小可表示为 $\frac{4}{3}m_2K_3N$;状态回放的平均时间复杂度为 $O(m_2\log_4 N)$。

综上所述,本文所采用的分层动态存储方式在空间复杂度上,相较于部件全信息存储以及部件状态同步存储均具有较大程度的减少。在时间复杂度上,本文的存储方式相较于其余两种存储方式,从线性级缩短至了对数级,运行效率显著提高。

2.3 轻量化日志生成与解析技术

在 DF 文件设计过程中,为了提高设计工具的容错率,需要对 DF 文件最新状态进行持久性日志备份,并在出现故障时进行加载解析。为了降低日志文件加载解析的复杂度,本文提出了一种基于属性分层的日志文件设计方法,实现了公有属性的统一管理,降低文件复杂度。此外,通过构建状态关联表对记录过程的数据进行筛选,剔除冗余数据,使得日志文件在保留了关键信息的同时,占用更少的内存空间,加快解析过程。

1) 属性分层

为了便于 DF 设计工具对日志文件的加载解析,本文设计了如表 2 所示的 XML 日志文件结构,该结构以 HistoryFile 为根节点,存储顶层日志信息。其子节点 FileState 记录 DF 文件的保存状态。FileState 之下是 Layer 节点,标识当前显示画面的图层号信息。

在 Layer 节点内部,本文通过构建以目标窗体部件属性为分类层 AttributeModify,以目标窗体部件及其状态 prop 为层内部结构的日志文件,将具有公有属性的窗体部件统一管理。相较于传统 DF 文件格式中以目标窗体部件为分类层,其属性以及状态为层内部结构的生成方式,能有效避免对相同目标窗体部件属性重复记录造成的文件内容冗余,从而实现日志文件的轻量化。

表 2 日志文件结构

Table 2 Structure of the log file

节点名称	父节点	节点描述
HistoryFile	NULL	历史日志描述
FileState	HistoryFile	DF 文件保存状态
Layer	FileState	图层号
AttributeModify	Layer	目标更新属性
prop	AttributeModify	目标窗体部件及状态

2) 状态筛选

为了避免记录与保存前初始窗体部件状态相同的更新状态,通过构建如图 5 所示的状态关联表,对这类状态内容进行剔除。

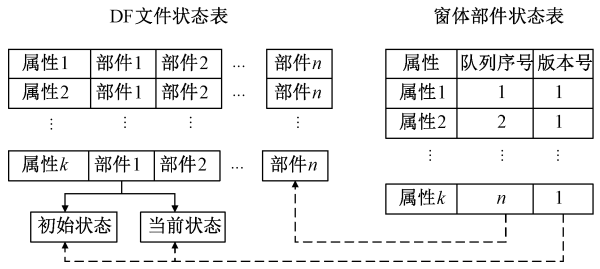


图 5 状态关联表

Fig. 5 State association table

状态关联表内部结构由 DF 文件状态表与窗体部件状态表联合维护,具体内容如下:

(1) DF 文件状态表

根据上述以目标窗体部件属性为分类层的策略,在 DF 文件状态表中同样构建以属性作为分类依据的部件集合,并对部件集合中的每个窗体部件添加目标更新属性的初始与当前最新状态集合。通过判断当前状态与初始状态是否一致,从而决定是否对当前状态进行剔除。

(2) 窗体部件状态表

在该表中,每个窗体部件的内部属性添加了额外的队列序号以及版本号。其中,队列序号对应窗体部件在 DF 文件状态表中对应属性集合中的位置,便于快速定位该属性的初始状态,从而判断与当前状态是否一致。版本号则对应 DF 文件状态表中状态集合内初始状态与当前状态的指向,根据其奇偶性,进行状态指向的动态更替,无需在状态保存后实时更改状态集合中所有的初始状态,从而降低 DF 文件状态表的维护难度。

为了对存在故障的 DF 文件进行数据恢复,需要获取故障前 DF 文件的状态,本文设计了如图 6 所示的日志文件解析流程。

首先,获取 FileState 节点属性为 unsaved 的片段。其次,根据其子节点的更新命令类型选用不同的解析方法,获

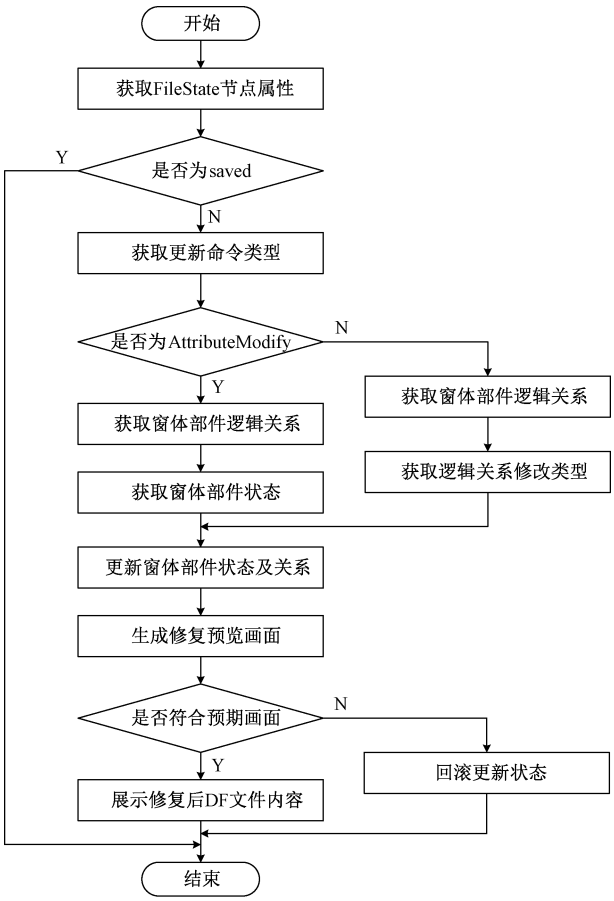


图 6　DF 文件恢复流程

Fig. 6　DF recovery process

取原始数据更新窗体部件。最后，生成当前 DF 文件的预览画面，实现修复内容的可视化。若不符合预期显示效果，可利用状态上下文管理器对更新状态进行回滚，仍显示上一版本的 DF 文件，保证了 DF 文件的完整性和一致性。

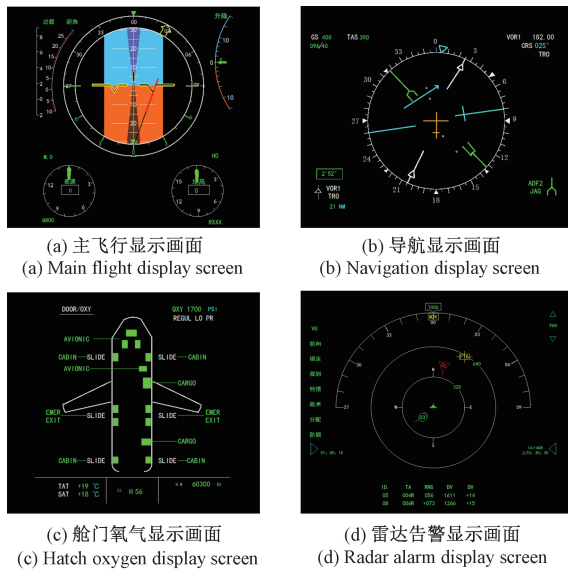
3　功能实例测试

本文通过搭建 DF 文件状态管理平台，利用如图 7 所示的主飞行显示画面、导航显示画面、舱门氧气显示画面以及雷达告警显示画面，对研究的关键技术进行测试验证。

1）存储结构性能测试

结合实际画面，对部件全信息存储、部件状态同步存储以及本文所设计的分层动态存储形式进行性能测试。实验中，通过编写自动化测试脚本，对主飞行显示画面中 142 个窗体部件的 1 595 个可修改属性均随机赋值 1、10、100 次，并对 DF 文件状态进行历史存储与回放，判断最终的显示画面是否与初始显示画面一致，从而验证本文所采用技术的可靠性。

在对主飞行显示画面中的窗体部件随机赋值后回放的过程中，选取其中一次测试结果，如图 8 所示。将俯仰平移容器的 TranslationY 下降 1 600，升降速率指针的



(a) 主飞行显示画面
(a) Main flight display screen

(b) 导航显示画面
(b) Navigation display screen

(c) 舱门氧气显示画面
(c) Hatch oxygen display screen

(d) 雷达告警显示画面
(d) Radar alarm display screen

图 7　初始测试画面

Fig. 7　Initial test screen

RotationAngle 更新至 60，表速指针与标签均改为 518，高度指针与标签均改为 6 116，分别经过上述 3 种存储方式的回放，其最终结果均符合图 8 的初始显示画面。

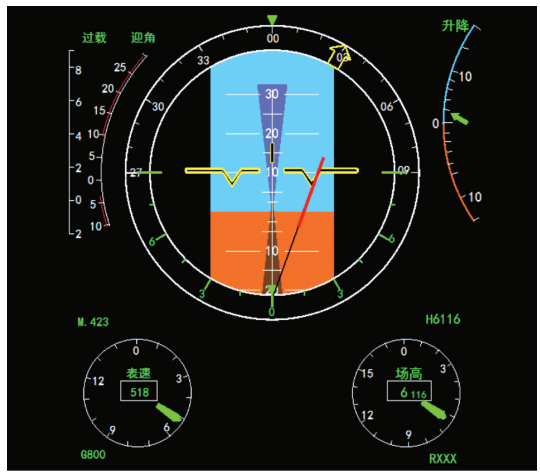


图 8　更新后的主飞行显示画面

Fig. 8　Updated main flight display screen

对 3 种存储方式在 DF 文件状态回放过程中的正确率进行统计，并针对单个属性修改 1 次、10 次以及 100 次时的平均耗时情况进行比较，如表 3 所示。由此可见，采用 3 种存储方式均能对 DF 文件状态进行正确的回放；在此基础上，本文所设计的存储结构在运行耗时方面具有更好的性能，并随着更新次数的增加，优势更加明显。

2）日志生成与解析测试

在设计导航显示画面、舱门氧气显示画面以及雷达告警显示画面时，每组利用统一更新命令对其所有窗体部件属性进行赋值，其中图 9(a)所示为采用传统方法生成的日志文件，图 9(b)所示采用本文方法生成的日志文件。

表 3 性能测试结果

Table 3 Performance test results

存储方式	实验次数	正确率/%	平均耗时/ms(1 次)	平均耗时/ms(10 次)	平均耗时/ms(100 次)
部件全信息存储	3 000	100	45	175	1 500
部件状态同步存储	3 000	100	8	8	43
分层动态存储	3 000	100	8	8	9

```
<?xml version="1.0" encoding="UTF-8"?>
<HistoryFile>
 <FileState state="unsaved">
  <Layer Id="1">
   <AttributeModify OldWidgetId="100">
    <prop name="WidgetIdent" value="100" />
    <prop name="Visible" value="True" />
    <prop name="PosY" value="-5848" />
    <prop name="PosX" value="-5898" />
   </AttributeModify>
   <AttributeModify OldWidgetId="101">
    <prop name="WidgetIdent" value="101" />
    <prop name="Visible" value="True" />
    <prop name="PosY" value="-5848" />
    <prop name="PosX" value="-5898" />
   </AttributeModify>
   <AttributeModify OldWidgetId="102">
    <prop name="WidgetIdent" value="102" />
    <prop name="Visible" value="True" />
    <prop name="PosY" value="-5848" />
    <prop name="PosX" value="-5898" />
   </AttributeModify>
```

(a) 传统方法
(a) Traditional method

```
<?xml version="1.0" encoding="UTF-8"?>
<HistoryFile>
 <FileState state="unsaved">
  <Layer Id="1">
   <AttributeModify name="PosY">
    <prop WidgetId="100" value="-5848" />
    <prop WidgetId="101" value="-5848" />
    <prop WidgetId="102" value="-5848" />
    <prop WidgetId="103" value="-5848" />
   </AttributeModify>
   <AttributeModify name="PosX">
    <prop WidgetId="100" value="-5898" />
    <prop WidgetId="101" value="-5898" />
    <prop WidgetId="102" value="-5898" />
    <prop WidgetId="103" value="-5898" />
   </AttributeModify>
   <AttributeModify name="Visible">
    <prop WidgetId="100" value="True" />
    <prop WidgetId="101" value="True" />
    <prop WidgetId="102" value="True" />
    <prop WidgetId="103" value="True" />
   </AttributeModify>
```

(b) 本文方法
(b) Proposed method

图 9 日志片段

Fig. 9 Log segments

针对生成的日志文件,统计每个机载显示画面所包含的窗体部件总数、属性总数以及分别采用两种方法生成的日志文件大小,其结果如表 4 所示。在同样的更新命令下,采用本文方法所设计的日志文件所占用的内存空间减少了 17.5%以上,满足轻量化的要求。

表 4 日志文件对比

Table 4 Log file comparison

画面类型	部件总数	属性总数	日志大小/KB(传统方法)	日志大小/KB(本文方法)
导航	37	429	27	22
舱门氧气	65	824	51	42
雷达告警	72	933	57	47

此外,针对故障 DF 文件的修复,需要对生成的日志文件进行加载解析。以舱门氧气显示画面为例,将其中 GpRectangle 部件的颜色均改为黄色后,直接强制退出软件不保存该 DF 文件。在上一版本的基础上,通过加载 XML 历史日志文件,对故障前的 DF 文件内容进行修复。同时,生成修复内容的可视化预览画面,如图 10 所示,用户可自行选择是否继续开发。由此可见,本文所采用的方法能有效地实现 DF 文件的版本修复,满足预期效果。

图 10 DF 文件修复结果

Fig. 10 DF repair result

4 结 论

本文针对座舱显示画面设计阶段 DF 文件状态管理问题,研究了 DF 文件的状态管理方法,并基于该方法研发了一套支持状态回溯以及日志追踪的 DF 文件状态管理平台。通过对窗体部件属性参数与功能接口进行建模,实现了窗体部件的统一管理。在此基础上,实现了设计过程中更新命令的高效存储,支持目标状态的快速定位。同时,通过生成轻量化日志文件,降低了数据备份的内存空间消耗,实现了 DF 文件版本的快速修复。该方法在座舱显示画面设计过程中,对 DF 文件状态实施了高效的管理,提高了开发效率,并保证了文件的完整性和可靠性。经过功能实例测试,本文所设计的 DF 文件状态管理方法可有效地实现各种功能需求,能够较好地满足座舱显示系统中机载画面的设计需求。

参考文献

[1] IQBAL M Z, SARTAJ H, KHAN M U, et al. A model-based testing approach for cockpit display systems of avionics [C]. 2019 ACM/IEEE 22nd International Conference on Model Driven Engineering Languages and Systems(MODELS), IEEE, 2019: 67-77.

[2] SARTAJ H, IQBAL M Z, KHAN M U. CDST: A toolkit

for testing cockpit display systems of avionics[C]. 2020 IEEE 13th International Conference on Software Testing, Validation and Verification(ICST), IEEE, 2020: 436-441.

[3] BARLADIAN B K, SHAPIRO L Z, DERYABIN N B, et al. Efficient rendering for the cockpit display system designed in compliance with the ARINC661 standard[J]. Programming and Computer Software, 2022, 48(3): 147-154.

[4] SARTAJ H, IQBAL M Z, KHAN M U. Testing cockpit display systems of aircraft using a model-based approach[J]. Software and Systems Modeling, 2021, 20: 1977-2002.

[5] ZHANG J, CAI Y. Design of ARINC661 data analysis software for airworthiness compliance verification [C]. 2018 IEEE CSAA Guidance, Navigation and Control Conference(CGNCC), IEEE, 2018: 1-6.

[6] 周贵荣,徐见源,马少博,等.大型客机航电系统综合集成关键技术综述[J].航空学报,2024,45(5):529956.
ZHOU G R, XU J Y, MA SH B, et al. Review of key technologies for avionics systems integration on large passenger aircraft [J]. Acta Aeronautica et Astronautica Sinica, 2024, 45(5): 529956.

[7] 许舒晨,孙永荣,冯悦,等.ARINC661 座舱显示系统离线检测技术研究[J].电光与控制,2019,26(5):77-80,89.
XU SH CH, SUN Y R, FENG Y, et al. On ARINC661 cockpit display system offline detection technology[J]. Electronics Optics & Control, 2019, 26(5): 77-80, 89.

[8] YANG W D, SHEN X, QIU Q CH, et al. An efficient approach for monitoring and analyzing real-time ARINC 661 events[C]. 2018 IEEE/AIAA 37th Digital Avionics Systems Conference(DASC), IEEE, 2018: 1-5.

[9] 陈斌.ARINC661 标准在直升机座舱显示系统设计的应用研究[J].新型工业化,2021,11(2):194-197.
CHEN B. Research on the application of ARINC661 in the design of helicopter cockpit display system[J]. The Journal of New Industrialization, 2021, 11(2): 194-197.

[10] 李回宝.民用飞机 ARINC661 仿真和监控试验台研究[J].科技创新与应用,2021,11(19):87-88,91.
LI H B. Research on civil aircraft ARINC661 simulation and monitoring testbed [J]. Technology Innovation and Application, 2021, 11(19): 87-88, 91.

[11] COUADAU F, DERVAUX N, FAYOLLAS C, et al. Automated testing of ARINC 661 cockpit display systems: factors to accelerate DO-178C certification [C]. 2023 IEEE/AIAA 42nd Digital Avionics Systems Conference(DASC), 2023: 1-7.

[12] 冯悦.ARINC661 航电总线数据可视化处理技术研究[D].南京:南京航空航天大学,2019.
FENG Y. Research on visualization technology of avionics bus data based on ARINC661[D]. Nanjing: Nanjing University of Aeronautics and Astronautics, 2019.

[13] 唐乐,孙永荣,许舒晨.ARINC661 地显系统 DF 文件设计平台的研究与实现[J].信息技术,2022,46(11):90-94.
TANG L, SUN Y R, XU SH CH. Research and implementation of platform for ground fire control display system definition file based on ARINC661[J]. Information Technology, 2022, 46(11): 90-94.

[14] 吴家秀,易子淳,吴之南,等.ARINC661 座舱显示图形设计与逻辑测试系统[J].航空计算技术,2022,52(6):102-106.
WU J X, YI Z CH, WU ZH N, et al. ARINC 661 cockpit display graphic design and logic test system[J]. Aeronautical Computing Technique, 2022, 52(6): 102-106.

[15] 薛霈.基于 ARINC661 的 Widget 库的设计与实现[D].西安:西安电子科技大学,2015.
XUE P. Design and implementation of the widget library based on ARINC661 [D]. Xi′an: Xidian University, 2015.

[16] 胡敏,裴晓芳,顾平月,等.边缘 AI 的霜雪识别系统设计与实现[J].电子测量技术,2021,44(1):142-149.
HU M, PEI X F, GU P Y, et al. Design and implementation of frost and snow recognition system for edge AI[J]. Electronic Measurement Technology, 2021, 44(1): 142-149.

[17] 吴家秀,张之江,易子淳,等.基于 Web 的 ARINC661 座舱显示内核封装与设计[J].工业控制计算机,2024,37(3):7-9.
WU J X, ZHANG ZH J, YI Z CH, et al. ARINC661 cockpit display kernel based on web[J]. Industrial Control Computer, 2024, 37(3): 7-9.

[18] 雷雨能,孙磊,魏正兵,等.基于 ARINC661 的火控系统 UA 设计[J].兵工自动化,2022,41(12):13-15.
LEI Y N, SUN L, WEI ZH B, et al. UA design of fire control system based on ARINC661[J]. Ordnance Industry Automation, 2022, 41(12): 13-15.

[19] WANG Y X, ZHANG SH, KONG D Q, et al. The design and implementation of primary flight display

components based on SCADE[C]. Piscataway: The Proceedings of 2021 IEEE 3rd International Conference on Civil Aviation Safety and Information Technology, IEEE, 2021: 470-474.

[20] 孔令波,陈茂胜,郑惠中.大型光学遥感卫星高可靠高性能综合电子系统设计[J].电子测量与仪器学报,2020,36(8):178-186.
KONG L B, CHEN M SH, ZHENG H ZH. Design of high reliability and high performance integrated electronic system for large optical remote sensing satellite[J]. Journal of Electronic Measurement and Instrumentation, 2020, 36(8): 178-186.

[21] 张开琦,刘晓燕,王信,等.基于微服务架构的中医信息服务系统设计与应用[J].电子测量技术,2020,43(3):1-5.
ZHANG K Q, LIU X Y, WANG X, et al. Design and application of TCM information service system based on micro-service architecture [J]. Electronic Measurement Technology, 2020, 43(3): 1-5.

[22] 谭旭,陈鹏飞,刘毅.基于 IEEE 1636.1 的自动测试结果数据管理软件设计与实现[J].国外电子测量技术,2020,39(1):151-156.
TAN X, CHEN P F, LIU Y. Design and implementation of automatic test data manager software based on IEEE 1636.1 [J]. Foreign Electronic Measurement Technology, 2020, 39(1): 151-156.

[23] 李笑瑜,孙永荣,赵科东. ARINC661 多显示器座舱系统中的 UA 画面处理技术[J].电光与控制,2018,25(6): 95-97,114.
LI X Y, SUN Y R, ZHAO K D. Picture processing technology of UA in multi-display cockpit system of ARINC661[J]. Electronics Optics & Control, 2018, 25(6): 95-97, 114.

[24] SINGH N K, AIT-AMEUR Y, MERY D, et al. Formal development of multi-purpose interactive application(MPIA) for ARINC 661[C]. International Workshop Formal Techniques for Safety-Critical Systems, 2020: 21-39.

[25] 聂飞,李健.基于 ARINC661 规范的航电分布式显控研究[J].计算机工程,2020,46(8):216-222.
NIE F, LI J. Research on distributed avionic display and control based on ARINC661 specification [J]. Computer Engineering, 2020, 46(8): 216-222.

[26] 李颖洁,王军,李柏瑞.座舱显示控制界面组件化设计方法研究[J].电光与控制,2022,29(3):105-109.
LI Y J, WANG J, LI B R. On componentized design of cockpit display and control interfaces [J]. Electronics Optics & Control, 2022, 29(3): 105-109.

作者简介

梁杭煜,硕士研究生,主要研究方向为航空机载显示技术。

E-mail:lhy898@nuaa.edu.cn

赵科东(通信作者),博士,讲师,主要研究方向为航空机载显示、视觉导航、测绘导航等技术。

E-mail:kd_zhao@nuaa.edu.cn