Chinastandards.org

ICS 25.040.30
CCS J 28

chinastandards.org - PREVIEW

NATIONAL STANDARD OF THE PEOPLE'S REPUBLIC OF CHINA

GB/T 33263-2016

Design specification for robotic software functional components

机器人软件功能组件设计规范

Issued on: December 13, 2016 **Implemented on: July 1, 2017**

Issued by: General Administration of Quality Supervision, Inspection and Quarantine;
Standardization Administration of the PRC

Preview: page 1 of 4 shown, 10 in the full document. Complete English PDF, instant download:
https://chinastandards.org/standards/gb-t-33263-2016

Chinastandards.org

Table of Contents

Chinastandards.org - PREVIEW

Foreword and introduction

This standard was drafted in accordance with the rules given in GB/T 1.1-2009.

It was proposed by the China Machinery Industry Federation and is under the jurisdiction of the National Technical Committee on Automation Systems and Integration Standardization (SAC/TC 159). The drafting organisations are Shanghai Jiao Tong University, the Beijing Institute of Automation for Machinery Industry and Siasun Robot and Automation.

The introduction sets out the case: as robots and other intelligent systems grow rapidly in complexity, robot market integration faces obstacles that the mature PC industry does not, because neither hardware modules nor software modules are compatible with one another.

The consequence it identifies is a great deal of repeated low-level development, a low degree of reuse, poor extensibility and a large waste of resources.

1 Scope

This standard specifies the terms and definitions, the abbreviations, the model of the robot functional component, the component state transitions and the integration method for robot software functional components.

It applies to the design and integration of software functional components for robot products.

The software half of the problem

The companion standards address hardware modularity; this one addresses software, and the software half is the harder of the two.

Preview: page 2 of 4 shown, 10 in the full document. Complete English PDF, instant download:
https://chinastandards.org/standards/gb-t-33263-2016

A hardware interface can be drawn: this bolt circle, this connector, these pins. A software interface has to specify behaviour, and behaviour is harder to pin down than geometry.

The observation in the introduction - that control software written for one robot cannot run on another - is not a complaint about laziness. It is what happens when software is written against a particular machine's assumptions about its own kinematics, timing and hardware.

Making it portable means separating what the component does from what it is running on, which is a discipline rather than a feature.

4 A component model

The document defines what a robot functional component is: its composition, its interfaces, and the model by which it is described.

The idea is the familiar one from software engineering generally - a unit with a declared interface and a hidden implementation - applied to a domain where it has been adopted unevenly.

What makes robotics awkward is that its components are not pure software. A component that reads a laser scanner or commands a joint is bound to a physical device with timing constraints and failure modes of its own.

So the model has to accommodate components that can fail because of the world rather than because of a bug, which is what the state machine is for.

5 The lifecycle is the useful part

Four states are defined: created, inactive, active and error, and fixing them is the most practically valuable thing in the document.

A framework that hosts components needs to be able to start them, stop them, and recover them without knowing what any of them do, and it can only do that if every component agrees on what those transitions mean.

The separation between created and inactive matters in particular: a component that has been constructed but not yet started can be configured and connected, and a system that brings up all its components before activating any of them fails in a much more tractable way than one that starts them as it builds them.

The explicit error state matters for the same reason. A component that fails and says so can be restarted or replaced; one that fails silently takes the system with it.

6 Integration, and the worked example

The integration method gives the sequence: design the component, create its model, debug it, test its communication, and integrate it into the system.

Preview: page 3 of 4 shown, 10 in the full document. Complete English PDF, instant download:
https://chinastandards.org/standards/gb-t-33263-2016

The communication test standing as a separate step reflects experience. A component that works correctly in isolation and communicates incorrectly is the common failure, and finding it during integration is far more expensive than finding it before.

Annex A runs the whole sequence through a worked example, from building the component model through the creator tool to a completed integration.

For a standard about software structure, the worked example is what makes it usable, since the abstract description of a component model reads the same whether or not anyone has ever built one.

chinastandards.org · PREVIEW

Preview: page 4 of 4 shown, 10 in the full document. Complete English PDF, instant download:
https://chinastandards.org/standards/gb-t-33263-2016