Entwurf und Realisierung einer Methode zur Exceptionbehandlung und Synchronisation in Echtzeitprogrammen

Author(s):
Maier, Georg Ernst

Publication Date:
1984

Permanent Link:
https://doi.org/10.3929/ethz-a-000315965 →

Diss. ETH Nr. 7583

Entwurf und Realisierung einer Methode zur Exceptionbehandlung und Synchronisation in Echtzeitprogrammen

ABHANDLUNG

zur Erlangung des Titels eines
DOKTORS DER TECHNISCHEN WISSENSCHAFTEN
der
EIDGENÖSSISCHEN TECHNISCHEN HOCHSCHULE ZÜRICH

vorgelegt von
GEORG ERNST MAIER
Dipl. El.-Ing. ETH
geboren am 31. Januar 1954
von Schaffhausen (SH)

Angenommen auf Antrag von:
Prof. Dr. W. Schaufelberger, Referent
Prof. Dr. A. Kündig, Korreferent

1984

23.10.84 W. Schaufelberger

Zusammenfassung

Eine Exception ist das Auftreten einer Bedingung, die es verunmöglicht, mit der Ausführung eines Programmes normal weiterzufahren. Es ist die Aufgabe eines Betriebssystems oder des Laufzeitsystems einer Programmiersprache, einen Mechanismus zur Verfügung zu stellen, welcher die Behandlung von Exceptions unterstützt.

Bekannte Konzepte schlagen vor, ein Programm in einzelne Bereiche zu gliedern, um Exceptions lokal behandeln zu können. Angepasst an eine blockstrukturierte Programmiersprache können diese Exceptionbehandlungsbereiche sequentiell oder verschachtelt angeordnet werden.

Dieser Ansatz wird auf parallele Programme übertragen: Dynamisch geschaffene Prozesse werden im selben Bereich ausgeführt, innerhalb welchem sie gestartet wurden. Sie können ihrerseits eigene Bereiche eröffnen, um Exceptions selbst zu behandeln, damit andere Prozesse nicht gestört werden. Beim Auftreten einer Exception wird die Ausführung des aktuellen Bereichs abgebrochen, und alle in diesem Bereich gestarteten Prozesse werden gestoppt.

Die Exceptionbehandlung muss in einem Echtzeitsystem zusammen mit der Synchronisation betrachtet werden, z.B. um Verklemmungen zu verhindern, weil Prozesse, die infolge von Exceptions abgebrochen werden, ihre Betriebsmittel nicht wieder freigeben. Es wird eine Verwaltung dynamischer Objekte vorgeschlagen, welche Prozesse, Synchronisationsdeskriptoren (z.B. Semaphoren) und Zugriffsrechte (z.B. Zugriff auf ein Betriebsmittel) unterscheidet. Ein Objekt wird am Ende des Bereichs, innerhalb welchem es geschaffen oder zugeteilt wurde, automatisch gelöscht resp. freigegeben.

Die Methode wurde im praktisch vollständig in Modula-2 codierten Modula-2 Echtzeitbetriebssystem MODEB V2 implementiert. Als Basis diente das Echtzeitbetriebssystemmodell des "Technical Committee No. 8 on Real Time Operating Systems" des "European Workshop on Industrial Computer Systems (EWICS)". Dieses Modell wurde um eine interne Schnittstelle erweitert, die als Basis zur Realisierung beliebiger Synchronisationskonzepte dient.

Nach der Spezifikation der Anwenderschnittstelle wurde der Entwurf in drei Schritten durchgeführt: Die allgemeine, implementationsunabhängige Einzelprozessorversion dient als Grundlage für die konkrete Realisierung auf einem PDP-11 Rechner und für die Uebertragung auf eng gekoppelte Multiprozessoren. Es wurde grosses Gewicht auf schrittweise Verfeinerung (top-down) und auf eine detaillierte Beschreibung des Vorgehens und der Lösung gelegt.

Einige Beispiele erläutern den Einsatz der vorgeschlagenen Methode in Anwendungs- und Systemprogrammen, und bestätigen die Bedeutung der Koppelung der Objektverwaltung an die Exceptionbehandlung.

Zeitmessungen auf einem PDP-11/45 Rechner zeigen, dass MODEB V2 für Datenerfassungen und Regelungen mit Abtastzeiten ab einigen Millisekunden eingesetzt werden kann.

Abstract

An exception is the occurence of a condition which prevents the flow of program execution to go on normally. It is the task of an operating system or of the run time system of a programming language to provide a mechanism supporting the handling of exceptions.

It has been proposed in other papers to subdivide a program to get the possibility to handle exceptions locally. According to block structered programming languages, these exception handling frames may follow each other or may be nested.

This model is generalized to parallel programs: Dynamically created processes are executed in the same frame within which they have been started. They may open their own frames to handle their exceptions themselves to not disturb other processes. Any exception aborts the execution of the current frame and all processes started within this frame are stopped.

In a real time system exception handling and synchronization must not be kept apart, e.g. to prevent deadlocks caused by processes not freeing their resources after they have been aborted by an exception. Therefore an object management is proposed which distinguishes between processes, synchronization descriptors (e.g. semaphors) and access rights (e.g. to resources). Objects are automatically deleted or returned at the end of the frame within which they have been created or granted.

The method has been implemented in the real time operating system MODEB V2 which is almost fully coded in Modula-2. The model of the "Technical Committee No. 8 on Real Time Operating Systems" of the "European Workshop on Industrial Computer Systems (EWICS)" has been used as a base and has been extended by an internal interface to support the realization of any synchronization concepts.

The specification of the application interface is followed by a three-fold design: A general implementation independant single processor version is developped which serves as a base for the real implementation on a PDP-11 computer and for the transfer to tightly coupled multi processors. Great importance has been attached to stepwise refinement (top-down) and to a detailed description of progress and solutions.

Some examples illustrate the use of the proposed method in application and system programs confirming the importance of object management coupled to exception handling.

Some time measurements on a PDP-11/45 computer show that MODEB V2 is applicable to data processing and control problems with sampling periods down to a few milliseconds.