Available online at www.sciencedirect.com
SCIENCE DIRECT
Fusion Engineering and Design 81 (2006) 1723–1727
Fusion Engineering and Design
www.elsevier.com/locate/fusengdes

JDAQ, the new TEXTOR data acquisition program

M. Korten, J.G. Krom *

Institut für Plasmaphysik[1], Forschungszentrum Jülich GmbH, EURATOM Association, D-52425 Jülich, Germany

Available online 26 May 2006

Abstract

During the last years many components of the TEXTOR data management system were re-engineered. This new system was successfully used to commission and subsequently to operate TEXTOR following the installation of the Dynamic Ergodic Divertor. This paper gives an overview of one of the main re-engineered components: JDAQ, the Java (or Jülich) Data AcQuisition system.

JDAQ is based on the design of, and the experiences with the previous TEXTOR data acquisition systems; it was aimed to be an open, distributed and scalable system. It has almost completely been written in the JAVA object-oriented programming language, reflecting many of the code patterns known from modern software engineering. JDAQ is designed as a four-tier layered system, which can be run on a single node or distributed over a TCP/IP network.

The TEXTOR operations during the last two years showed the advantage of a highly flexible, platform independent and modular development. The majority of our diagnostic subsystems have been moved to JDAQ and have been reliably operated in the TEXTOR experimental campaigns.

Keywords: Fusion research; Data acquisition; Java; Object-oriented; Patterns-based; Software engineering

1. Introduction

TEXTOR is the fusion research tokamak of the Research Centre Jülich. It is operated by an international community: the Trilateral Euregio Cluster (TEC) consisting of Forschungszentrum, Jülich (Germany), FOM-Rijnhuizen (The Netherlands) and ERM-Brussels (Belgium).

During the late 1990s, the TEC community planned to enhance the TEXTOR device with the dynamic ergodic divertor (DED) [1]. This divertor was installed during a major shutdown of TEXTOR operations in 2002 and 2003.

This shutdown period gave us also the opportunity to upgrade the control, data acquisition and data management systems in use around TEXTOR. These upgrades were influenced by earlier work in the European Union Framework programs REMOT, DYNACORE [2] and related projects [3,4].

This paper describes the main new software development: JDAQ: The Java coded diagnostic data acquisition system.

* Corresponding author. Tel.: +49 2461 615451; fax: +49 2461 615452.
E-mail address: j.krom@fz-juelich.de (J.G. Krom).
[1] Partner in the Trilateral Euregio Cluster (TEC).

0920-3796/$ – see front matter © 2006 Elsevier B.V. All rights reserved.
doi:10.1016/j.fusengdes.2006.04.033

2. JDAQ data acquisition

One of the major developments around TEXTOR in the last few years is the new data acquisition system: JDAQ, the Jülich (or Java) Data Acquisition. It replaces the previous system which, though successful in its design concepts, had turned out to be an increasingly serious obstacle for further extensions. This was due to the legacy implementation (VAX Macro Assembler, FORTRAN) and hardware dependencies (MicroVax Q-bus, CAMAC). Embarking from the proven design principles of the old system, JDAQ aimed to be a distributed and scalable new design.

JDAQ has been written in the JAVA object-oriented programming language to benefit from modern software technologies and a rich literature of code and design patterns. This open software architecture enabled the support of new data acquisition hardware and has lead to a fast replacement of the old acquisition system.

JDAQ uses the JAVA Standard Development Kit, which is available free of charge and makes the core of JDAQ platform independent. It uses JAVA native interfacing to connect to platform specific hardware drivers which are usually available from the data acquisition instrumentation vendors. JDAQ is independent of any other third-party commercial products. Beyond those components that directly use the mentioned hardware drivers, JDAQ can be employed on all platforms that are supported by JAVA. It is being used mainly on PCs with Microsoft Windows (XP, 2000 and NT), but has been shown to work under Linux; at TEXTOR the storage component routinely runs under IBM-AIX.

JDAQ is, like our previous and many comparable systems, organized to support autonomously operating diagnostic subsystems. These subsystems represent, in the JDAQ case, organizational units related with a TEXTOR diagnostic, or with a group of TEXTOR diagnosticians. These subsystems are not, like in some comparable acquisition systems, separate computers, but are represented by objects in an object-oriented program.

These subsystems can run completely independent from each other or synchronized under control of the central TEXTOR timing. The TEXTOR timing system supports an interactive mode, that provides complete local control for testing and diagnostic development, and an automatic mode, for data taking during TEXTOR discharge. In this mode all subsystems make use of the TEXTOR wide synchronized clocks and trigger events.

2.1. JDAQ Topology

JDAQ is designed as a multi-tier, network distributed system. This topology has been depicted in Fig. 1. The relevant tiers are:

1. DIALOG: The graphical user interface tier. These are intuitive displays that allow for status monitoring, control and surveillance of the all JDAQ subsystems as a whole, or specific selected ones in detail. Sensitive control and parameter manipulations are authentication protected.
2. DISPATCH: The diagnostic subsystem tier. JDAQs central control of all supported subsystems. Interacts with a configuration database, interactive dialog user sessions and the device tier. Dispatch executes individual subsystem control by delegating module specific hardware commands to the device tier. Depending on a pre-selected control mode, dispatch acts either autonomously, or driven and synchronized by the TEXTOR timing system.
3. DEVICE IO: The data acquisition tier. Device servers provide the actual I/O transactions to any supported data acquisition hardware. A device server typically handles the hardware behind the bus-controller of a specific type, i.e. CAMAC controllers.
4. ARCHIVE: The JDAQ data storage tier: This tier is responsible for storing and archiving all relevant data. This data is then made accessible by the end-users via the TEC Web-Umbrella mechanism [5].

This design allows the different JDAQ activities to be deployed on different computers, which in turn improves the scalability of the whole. This feature has been used in TEXTOR operations: some subsystems took far longer than others to complete their acquisition cycle, with some judicious redistributing of the JDAQ services over the available computers we managed to significantly improve the overall performance.

2.1.1. The JDAQ DIALOG server

The user interface to JDAQ is provided through the JDAQ dialog server, which can be run from any computer with support for a JAVA virtual machine and