COTS Components and DB Interoperability

Radmila Juric and Ljerka Beus-Dukic

Cavendish School of Computer Science, University of Westminster,
115 New Cavendish Street, London W1W 6UW, United Kingdom
R.Juric@wmin.ac.uk, L.Beus-Dukic@wmin.ac.uk

Abstract. The paper addresses the specific issue of interoperability in heterogeneous databases (DBs) and the possible use of COTS components that may alleviate the DB interoperability problem. A component-based software Architectural Style (AS) for interoperable DBs has been used, and an example of its application given, to identify which role the COTS components may play when populating the architecture. We discuss the characteristics of such COTS components and advocate that such COTS components should be developed with a specific component platform in mind, interoperate within a certain context, and adhere to constraints of our AS.

1 Introduction

DB systems have experienced vigorous change in the last decade because of the strong pull of commercial applications and the incessant push of technology and research advances. Heterogeneity and distribution have become main characteristics of DB systems, placing the question of interoperability in the focus of interests in both academia and industry. The increasing complexity of software systems and infrastructures, have pushed forward component-based software engineering practices, aiming to develop software from pre-produced reusable software components. Capitalising on third-party expertise and synthesising component technologies with commercial-of-the-shelf (COTS) components might bring new answers to problems of interoperability when building today's software systems.

This work merges long-term research in the DB interoperability field with component technologies and COTS components. We use a component based AS for interoperable DBs and identify the characteristics of COTS components that may populate the AS and alleviate the DB interoperability problem.

There are many interoperability perspectives in today's software systems, ranging from interoperability across domains and systems, to software architecture composability of components and their interactions. In our definition of DB interoperability [12], interoperable DBs exhibit communication and use of each others' data and functionality, despite their heterogeneities. Hence the works on components' interoperability such as [11], or on interoperability between software systems and system of systems from [14], or on software multi-operability from [23], are not discussed in this paper. Our work is *solution-specific*, i.e. it addresses a specific issue of DB interoperability and the use of COTS components. This is close to works of [33, 22], where

X. Franch and D. Port (Eds.): ICCBSS 2005, LNCS 3412, pp. 77–89, 2005.

COTS are used for addressing heterogeneity and interoperability in GIS and web information systems. Our AS for interoperable DBs fits within the Universal Systems layer for enterprise-wide shared system of the Levels of Information System Interoperability (LISI) model [21]. We also conform to the Seamless Sharing of Information of the structured data interchange degree in the NC3TA Reference Model for Interoperability (NMI) [25].

Tightly coupled with the term interoperability is the concept of integration. Integration is very important in COTS-based systems development and the idea of "focusing on integration to achieve interoperability" is at the core of the COTS-Based Systems Initiative Group tasks [10]. However, integration per se does not always bring interoperation in multiple DBs, because it affects individual DB's autonomy and evolution, and might have an impact on their distribution [12,16]. A technical initiative [6] points out that many heterogeneous systems might be perceived as integrated, but from the perspectives of their individual constituent elements, they actually inter-operate with each other. Consequently, we use both terms carefully and leave discussion on using integration and interoperation interchangeably outside the scope of this paper.

Section 2 comments on related research. Section 3 defines our layered reference model and the building blocks of the AS for interoperable DBs. Section 4 introduces a motivating example, designed as an EJB [28] application. Section 5 outlines characteristics of COTS components that can partially populate our AS. Section 6 gives conclusions and a brief outline of our future work.

2 Related Background

The numerous works related to the semantics of DB interoperability, which have intrigued researchers from the DB community since the early 1990s, range from

(a) migration between various DB systems [29] and
(b) multidatabase and federated architectures [13] to
(c) the mediator paradigm [35] which has culminated in many research projects from the 90s, such as [7], some of its applications on the Internet [26] and their counterparts emerging from industry [31].

Each of these approaches to DB interoperability, and particularly various levels of integration have drawbacks [12,16], and today's trend is to

i. allow the individual DB to evolve naturally within its own environment, and
ii. build/offer services, i.e. to use service-oriented technologies that will provide transparent facilities across different DB systems.

Thus current solutions should move away from known ideas of integration of data/applications, centralisation of data structures, federation architectures and multidatabase languages when addressing the heterogeneity of data centric applications and their interoperability. We have used a layered software architecture as defined in [4] and have proposed a component-based AS for interoperable DBs, given in Fig. 1 and 2, which should allow interoperation amongst multiple DBs and a certain level of evolution and autonomy of individual DBs [12,16]. Our architectural model has been