A Framework for Building Collaboration Tools by Leveraging Industrial Components

Du Li, Yi Yang, James Creel, and Blake Dworaczyk

Center for the Study of Digital Libraries (CSDL)
Texas A&M University
lidu@csdl.tamu.edu

Abstract. Groupware applications allow a distributed group of human users to work apart together over a computer network. They are difficult to develop due to the needs to suit a range of collaboration tasks that are often with diverse and evolutionary requirements. To address this problem, we propose a new framework in which shared data components conforming to a well-defined interface can be dynamically plugged in for flexible sharing, and a simple transformation tool is provided such that the myriad of industrial collaboration-transparent components can be transformed into shared components. The validity of our framework is evaluated by building a suite of typical collaboration tools such as group editors. Under our framework, most components in the Java Development Kit (JDK) can be transformed automatically for prototyping collaboration tools. With minimal manual work, those tools can be adapted to achieve advanced flexibility, e.g., data and control components can be bound dynamically to switch control protocols.

1 Introduction

The support of distributed collaboration has been penetrating modern computer systems in the past decade. This is evidenced by the increasing popularity of standalone groupware applications, such as instant messengers, multiplayer games, and electronic meeting systems, and groupware features integrated in group productivity tools, such as many recent programming environments and office products. In general, groupware needs to win a critical mass of users to be a success [5]. This often requires that they be sensitive to the needs of a variety of user groups and collaboration tasks. However, the needs of users and tasks often differ and evolve over time [18].

One way to address this challenge is to provide customizable groupware tools which allow users to set parameters that match their particular preferences. However, this approach often results in bloated systems that are difficult to maintain and evolve, while the level of flexibility is limited to what are foreseen and parameterized at design time. If groupware tools are developed ad hoc as separated applications, reusability and extensibility will be limited.

An alternative approach is to provide a reusable groupware infrastructure for users to easily prototype desired groupware tools as the needs emerge. Conceptually, this would make the practices of groupware development more systematic,
resulting in lower engineering and re-engineering costs, well-architected systems, and a consistent look and feel across different tools.

Recent works such as [7,17,22] represent a trend to integrate the practices of groupware development and the practices of component-based development (CBD) in software engineering [20]. While it is well understood that groupware infrastructures must provide suitable programming abstractions [16], open issues include how to model components in groupware and how to fill the gap between groupware components and current industrial components that do not observe custom groupware component models.

We propose a novel framework called EXEC (an Evolvable and eXtensible Environment for Collaboration) that addresses these two issues. First, it includes a groupware component model, which decomposes a range of groupware applications into shared data components and control components. Based on this model, data and control components can be developed as independent libraries, and the infrastructure provides services to dynamically compose them at run time. Second, it includes a transformation tool which is able to transform existing collaboration-transparent components (e.g., JavaBeans) such that they conform to our custom component model and can be plugged into our platform for sharing. This way our framework can leverage the myriad of industrial components for fast-prototyping flexible groupware applications.

In the next section, we review related works to motivate the proposed framework. Then we elaborate the proposed framework in Sections 3 and 4. Next, Section 5 evaluates the framework by concrete design examples. Section 6 summarizes contributions and future directions.

2 Related Work

There are generally two approaches to developing groupware: The first is collaboration awareness, in which the system is designed to support multiuser interaction, possibly aided by an infrastructure (toolkit) with reusable services [10]. The second approach is collaboration transparency, in which a (reusable) collaboration-aware infrastructure is provided such that an existing single-user application is shared for multiuser interaction without modifying its source code [12]. Recent component-based approaches combine the advantages of both [17].

Collaboration Awareness. There are a plethora of specialized groupware applications, e.g., [19]. Over the years, many of the collaboration services such as those for group communication, consistency control and session management are identified as common to a variety of groupware applications. Reusable services are provided in groupware toolkits such as DistEdit [10], Suite [3], and GroupKit [16] such that new groupware applications can be built more easily.

Tooltkits generally provide custom programming abstractions with embedded consistency control protocols, and assume that groupware applications are developed based on these abstractions. For example, when used for building a group editor, Suite and GroupKit both require that the shared data be manually constructed in their custom data structures (e.g., sequences [3] and dictionaries.