Available online at www.sciencedirect.com
ScienceDirect
Electronic Notes in Theoretical Computer Science
ELSEVIER
Electronic Notes in Theoretical Computer Science 168 (2007) 175–190
www.elsevier.com/locate/entcs

Component-Based Specification of Collaborative Objects

Abdessamad Imine[1]

LORIA, INRIA - Lorraine
Campus Scientifique, 54506 Vandœuvre-Lès-Nancy Cedex, France

Abstract

A collaborative object represents a data type (such as a text document or a spreadsheet) designed to be shared by multiple geographically separated users. In order to improve performance and availability of data in such a distributed context, each user has a local copy of the shared objects, upon which he may perform updates. Locally executed updates are then transmitted to the other users. So, the updates are applied in different orders at different copies of the collaborative object. This replication potentially leads, however, to divergent (*i.e.* different) copies. The Operational Transformation (OT) approach provides an interesting solution for copies divergence. Indeed, every collaborative object has an algorithm which transforms the remote update according to local concurrent ones. But this OT algorithm needs to fulfill two conditions in order to ensure the convergence. Proving the correctness of OT algorithms is very complex and error prone without the assistance of a theorem prover. In the present work, we propose a compositional method for specifying complex collaborative objects. The most important feature of our method is that designing an OT algorithm for the composed collaborative object can be done by reusing the OT algorithms of component collaborative objects. By using our method, we can start from correct small collaborative objects which are relatively easy to handle and incrementally combine them to build more complex collaborative objects.

Keywords: CSCW, groupware systems, component-based design, formal methods

1 Introduction

Distributed collaborative systems allow two or more users (sites) to simultaneously manipulate objects (*i.e.* text, image, graphic, etc.) without the need for physical proximity and enable them to synchronously observe each other's changes. In order to achieve an unconstrained group work, the shared objects are *replicated* at the local memory of each participating user. Every operation is executed locally first and then *broadcasted* for execution at other sites. So, the operations are applied in different orders at different *replicas* (or copies) of the object. This potentially leads to *divergent* (or different) replicas – an undesirable situation for replication-based collaborative systems [10].

[1] Email: imine@loria.fr

doi:10.1016/j.entcs.2006.08.027

Operational Transformation (OT) is an approach which has been proposed to overcome the divergence problem, especially for building real-time groupware [1,8]. This approach consists of an algorithm which transforms an operation (previously executed by some other site) according to local concurrent ones in order to achieve convergence. It has been used in several group editors [1,6,8,7,11,9], and it is employed in other replication-based groupwares such as a generic synchronizer [5]. The advantages of this approach are: (i) it is independent of the replica state and depends only on concurrent operations; (ii) it enables an unconstrained concurrency, *i.e.* no global order on operations is required; (iii) it ensures a good responsiveness in real-time interaction context. However, if OT algorithms are not correct then the consistency of shared data is not ensured. Thus, it is critical to verify such algorithms in order to avoid the loss of data when broadcasting operations. According to [6], the OT algorithm of every collaborative object needs to fulfill two *convergence conditions* $TP1$ and $TP2$ that will be detailed in Section 2. Finding such an OT algorithm and proving that it satisfies $TP1$ and $TP2$ is not an easy task. This proof is often difficult – even impossible – to produce by hand and unmanageably complicated. In [4], we proposed a formal framework for modeling and analyzing the OT algorithms with algebraic specifications. For checking the convergence conditions we used a theorem prover. Using our formal approach we have detected bugs in well-known OT algorithms.

Until now the OT approach has been used to only deal with simple collaborative objects, such as a string object. When we consider a complex object (such as a filesystem or an XML document that are composite of several primitive objects) the formal design of its OT algorithm becomes very tedious because of the large number of updates and synchronization situations to be considered if we start from scratch. As continuation of [4], we propose in the present work a compositional method for specifying complex collaborative objects. The most important feature of our method is that designing an OT algorithm for the composed collaborative object can be done by reusing the OT algorithms of component collaborative objects. By using our method, we can start from correct small collaborative objects (*i.e.* they satisfy convergence conditions) which are relatively easy to handle and incrementally combine them to build more complex collaborative objects that are also correct.

This paper is organized as follows: in Section 2 we give the basic concepts of the OT approach. The ingredients of our formalization for specifying the collaborative object and OT algorithm are given in Section 3. In Section 4, we present two constructions for composing collaborative objects in algebraic framework. Finally, we give conclusions and present future work.

2 Operational Transformation Approach

Due to high communication latencies in wide-area and mobile wireless networks the replication of collaborative objects is commonly used in distributed collaborative systems. But this choice is not without problem as we will see in next sub-section.