ETH zürich

ETH Library

On document-centered mathematical component software

Doctoral Thesis

Author(s):
Weck, Wolfgang

Publication date:
1996

Permanent link:
https://doi.org/https://doi.org/10.3929/ethz-a-001710334

Rights / license:
In Copyright - Non-Commercial Use Permitted

This page was generated automatically upon download from the ETH Zurich Research Collection.
For more information, please consult the Terms of use.

Diss. ETH ex. B

On Document-Centered Mathematical Component Software

A dissertation submitted to the
Swiss Federal Institute of Technology Zurich (ETH Zürich)
for the degree of
Doctor of Technical Sciences
presented by

Wolfgang Weck
Dipl. Informatik-Ing. ETH
born December 7, 1964
citizen of Germany

CatE

accepted on the recommendation of
Prof. G. H. Gonnet, examiner
Prof. N. Wirth, co-examiner

Diss. ETH No. 11817
1996

Zusammenfassung

Es ist erstrebenswert, Software aus Komponenten zusammenzustellen, die unabhängig voneinander hergestellt und vermarktet werden. Solche Komponenten müssen innerhalb eines Frameworks entwickelt werden, um später kombiniert werden zu können. Dieses Framework hängt vom Anwendungsgebiet der Software ab.

Parallel zur komponentenorientierten Softwarearchitektur benötigt man ein erweiterbares Benutzerinterface. Dokumente sind eine gute Basis dafür. Einzelne Softwarekomponenten werden als Editoren für Dokumente oder für Teile davon präsentiert. Solche Software wird als dokumentenzentriert bezeichnet.

In dieser Dissertation wird ein Framework für mathematische Komponentensoftware präsentiert und ein dokumentenzentriertes Benutzungsmodel entwickelt. Beide sind eine Fallstudie für das Design dokumentenzentrierter Komponentensoftware im allgemeinen, und zugleich ein konkreter Vorschlag im Bereich mathematischer Software.

Durch die Benutzung einer allgemeinen Programmierumgebung anstelle eines Systems für mathematische Software, können existierende Dokumenteneditoren weiter benutzt werden. Die Möglichkeit zur Kombination mit weiteren Komponenten, z.B. für Hypertext, ist ein wichtiger Schritt in Richtung interaktiver Lehrbücher.

Die wesentlichen Beiträge der Dissertation sind folgende:

Mathematische Software sollte in drei Richtungen erweiterbar sein: Typen mathematischer Ausdrücke, Algorithmen und Editoren. In unserer Architektur werden die mathematischen Ausdrücke zu Verbindungselementen zwischen berechnender Software und den Editoren. Mathematische Ausdrücke in Dokumenten dürfen nicht durch Automatismen verändert

werden. Es darf beispielsweise keine Normalform erzwungen werden, wie dies viele auf das Rechnen gerichtete Systeme tun. Da gleichzeitig dieselben Ausdrücke für Berechnungen benutzt werden sollen, müssen sie mathematische Objekte repräsentieren, nicht nur Grafiken. Daher ist eine Abstraktion mathematischer Ausdrücke, die beiden Aspekten gerecht wird, fundamental in dieser Arbeit.

In erweiterbarer mathematischer Komponentensoftware müssen Ausdrücke als unveränderbare, gerichtete, azyklische Graphen dargestellt werden. Es wird ein Design Pattern für solche Graphen eingeführt. Dieses erzwingt die nötigen Restriktionen, ist zugleich aber bequem zu benutzen.

Compound-Documents können als Datenspeicher benutzt werden. Sie haben einige Vorteile gegenüber solchen, die auf Zuweisungen an Namen beruhen. Spezielle Dokumenten-Parts repräsentieren mathematische Ausdrücke. Eine neue Idee ist, solche Dokumenten-Parts auch zu benutzen um zukünftige Resultate zu repräsentieren, die parallel berechnet werden. Ein weiteres neues Konzept ist Dokumenten-Parts in ihrem Kontext zu interpretieren. Damit lässt sich ein einfacher Formeleditor und ein Interpreter für Skripte herstellen.

Abstract

The idea of composing software from components, which are created and marketed independently of each other is appealing. To be composable later, such components need to be developed within some framework. This framework depends on the software's domain of application.

In parallel to the component-oriented software architecture an extensible user interface is needed. Documents are a good basis for this. Individual software components are presented as editors of documents or parts thereof. Such software is called document-centered.

In this thesis a framework for mathematical component software is presented and a document-centered user model is developed. Both are a case-study in the design of document-centered component software in general and, at the same time, a concrete suggestion in the domain of mathematical software.

Using a general purpose programming environment instead of a mathematical computation system allows us to reuse existing document editors. Composability with further facilities, like hypertext support, is an important step towards interactive textbooks.

The main contributions of the thesis are the following:

Mathematical software should be extensible in three directions: expression types, algorithms, and editors. In our architecture, expressions are used to link computational software and editors. Expressions within documents must not be changed automatically. For instance, the software must not enforce a normal form, like many systems focused on computing would do. Since at the same time, the same expressions shall be used for calculating, they must represent mathematical objects, not just graphics.

Therefore, an abstraction of expressions, allowing to deal with both issues, is fundamental to this work.

With extensible mathematical component software, expressions must be stored as immutable, directed, acyclic graphs. We introduce a new design pattern for such graphs. This design pattern enforces the necessary restrictions, yet clients can use it conveniently.

Compound documents can be used as data repositories, having certain advantages over such based on name bindings. Special document parts represent expressions. A new idea is to use such document parts also to represent upcoming results, being computed concurrently. Another new concept is to interpret document parts within their context. This is used to create a simple expression editor and a script interpreter.