ETH zürich

ETH Library

Eine Domänen-Architektur zur Konfiguration von wiederverwendbaren Software-Komponenten für die Lösung nichtlinearer Netzgleichungen

Doctoral Thesis

Author(s):
Bosshart, Peter

Publication date:
1997

Permanent link:
https://doi.org/https://doi.org/10.3929/ethz-a-001847034

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 Nr. 12317

Eine Domänen-Architektur zur Konfiguration von wiederverwendbaren Software-Komponenten für die Lösung nichtlinearer Netzgleichungen

ABHANDLUNG
zur Erlangung des Titels
DOKTOR DER TECHNISCHEN WISSENSCHAFTEN
der
EIDGENÖSSISCHEN TECHNISCHEN HOCHSCHULE
ZÜRICH

vorgelegt von
PETER BOSSHART
Dipl. El.-Ing. ETH
geboren am 15. Juli 1955
Bürger von Zürich (ZH)
und Wittenwil (TG)

Angenommen auf Antrag von
Prof. Dr. R. Bacher, Referent
Prof. Dr. R. Weiss, Korreferent

Zürich 1997

Kurzfassung

Viele Applikationen im Bereich der elektrischen Energieübertragung führen auf ein System von simultanen, nichtlinearen Gleichungen $\vec{g}(\vec{x}, \vec{p}) = 0$ ($\vec{x}$: Unbekannte, $\vec{p}$: Parameter), die nach den Unbekannten $\vec{x}$ aufzulösen sind. Diese Gleichungen beschreiben das nichtlineare Verhalten miteinander verbundener Entitäten.

Dies ist kein neues Problem. Der Beitrag der vorliegenden Arbeit besteht jedoch darin, für diese Klasse von Problemen mit Mitteln des Software-Engineerings und aktuellen Erkenntnissen aus der Netzmodellierung eine wiederverwendbare, generische Lösung zu beschreiben, die leicht an neue Anforderungen angepasst werden kann.

Diese Lösung besteht konkret aus den Diensten Netzaufbau, Netzberechnung und der Darstellung von Resultaten. Diese Dienste werden in einheitlicher Art als Leistungen aufgefasst, die sich aus einem variablen und einem innerhalb der Problemklasse konstanten Anteil zusammensetzt. Diese Struktur ist in einem in der vorliegenden Arbeit betrachteten Prototypen konkretisiert worden.

Der Kern des Prototypen besteht aus einem objekt-orientierten Framework. Es eignet sich für diese Aufgabenstellung, weil sich damit ein generisches Modell der betrachteten Problemklasse implementieren lässt. Dieses Domänenmodell stellt ein wesentliches Ergebnis der vorliegenden Arbeit dar.

Es hat sich herausgestellt, dass das Verhalten eines Netzes durch Instanzen zweier generischer Bausteintypen, **Ntor** und **Summengeber**, modellierbar ist. Instanzen dieser Typen werden stets über Anschlüsse miteinander verbunden. Dabei wird der Anschluss als abstrakter Begriff aufgefasst. Die dafür gängige Vorstellung einer elektrischen Klemme ist nur ein wichtiger Spezialfall.

Das **Ntor** ist eine Entität, die üblicherweise mit einem physikalischen Betriebsmittel, wie z.B. einem Generator, einer Last oder Leitung assoziiert wird. Das Ntor ist jedoch von allgemeinerer Natur. Es ist mit ihm möglich, auch eine Fehlerbedingung, wie z.B. einen Kurzschlussfall zu modellieren. Das Ntor stellt einen endlichen Automaten mit einem Verhalten $\mathcal{BN}_\sigma(\mathbb{V}_\sigma, \mathbb{P}_\sigma, \mathbb{E}_\sigma, \mathbb{T}_\sigma)$ für jeden Zustand σ dar. Die Menge $\mathbb{E}_\sigma$ enthält alle Gleichungen, die mit Hilfe der Variablen $\mathbb{V}_\sigma$ und den Parametern $\mathbb{P}_\sigma$ gebildet werden. Die Menge $\mathbb{T}_\sigma$ spezifiziert den endlichen Automaten durch die Angabe von Zuständen und

Übergangsbedingungen.

Die **Summengeber** greifen auf Variable in den Ntoren zu, und formulieren damit weitere Gleichungen, um z.B. die Topologie eines Netzes auszudrücken.

Den Bausteintypen des Domänenmodells entsprechen spezielle Klassen des Frameworks, die dazu angelegt sind, für einen bestimmten Anwendungsfall noch konkretisiert zu werden. Dazu wird ein Mechanismus vorgestellt, der den variablen Anteil eines Dienstes in kontrollierter Weise im Framework adaptiert. An dieser Stelle kommt ein symbolisches Rechenprogramm (Maple V) zum Einsatz, das aus einer anwendungsfreundlichen Hochsprachen-Spezifikation zuverlässig den korrekten Quelltext für die zu konkretisierenden Teile des Frameworks erzeugt.

Alle diese Elemente, wie Framework, Spezifikation und Adaption der generischen Teile des Frameworks werden innerhalb einer übergeordneten Domänen-Architektur koordiniert.

Der Nutzen der in dieser Arbeit vorgeschlagenen Vorgehensweise ist eine erhebliche Einsparung des Entwicklungsaufwandes innerhalb dieser Klasse von Simulationsproblemen.

Abstract

Many applications lead to systems of simultaneous nonlinear equations $\vec{g}(\vec{x}, \vec{p}) = 0$ ($\vec{x}$: unknowns, $\vec{p}$: parameters) which have to be solved for the unknowns $\vec{x}$. It is assumed that these equations describe non-linear behavior related to interconnected entities. This is not a new problem. However the contribution of this thesis is a description of a generic solution for a class of problems by means of software-engineering and an actual approach of network modeling. The result is a reusable asset which can be easily adapted to changes of specifications.

The proposed asset offers three fundamental services: Instantiation of network building objects, simulation (calculation) and representation of results. These services are modeled in a unified manner. Each service consists of a variable and a constant part within the boundary of the problem. This concept has been implemented in a prototype, which is described in the thesis.

The core of the prototype is an object-oriented framework. It is suited for the task, because it is capable to represent a generic design for the class of problems mentioned above. A domain model containing the commonalities of these problems has been found, which is an important result of this thesis.

A typical application was analyzed to figure out an object-oriented model of this domain. It turned out that a behavior of a network can be modeled by instances of two generic types **nport** and **sumsetter**. These instances are always connected by **ports**. Thereby the port is seen as an abstract term. Its common association to an electrical terminal is only an import special case.

The **nport** is the entity typically associated with an electrical element such as generator, load, line etc. The nport, however, is more general allowing to model for example fault conditions, such as a short circuit. The nport is a finite state machine with a behavior $\mathcal{BN}_\sigma(\mathbb{V}_\sigma, \mathbb{P}_\sigma, \mathbb{E}_\sigma, \mathbb{T}_\sigma)$ for each state σ. The set $\mathbb{E}_\sigma$ contains all equations formulated in terms of local variables $\mathbb{V}_\sigma$ and parameters $\mathbb{P}_\sigma$. $\mathbb{T}_\sigma$ defines the transitions and their conditions leading to next states.

A **sumsetter** accesses variables located in nports to create further equations, typically expressing incidence relations within networks. A sumsetter contains instances of sum types and a set $\mathbb{A}$ of n connected ports.

The domain model was implemented as a system of coupled classes which establish the object-oriented framework. Thereby the entities of the domain model, nport and sumsetter, are special classes, which have to be adapted to concrete applications. A mechanism is proposed to adapt the framework in controlled way. At this place a symbolic calculation system (Maple V) is applied to generate correct source code of the adapted parts of the framework from a user-friendly, high-level language specification.

All these elements, such as framework, specification and adaptation of the generic parts of the framework are coordinated within a domain architecture.

The payoff of this approach is a high reduction in the effort to develop, maintain and expand software in the domain of non-linear network equations.