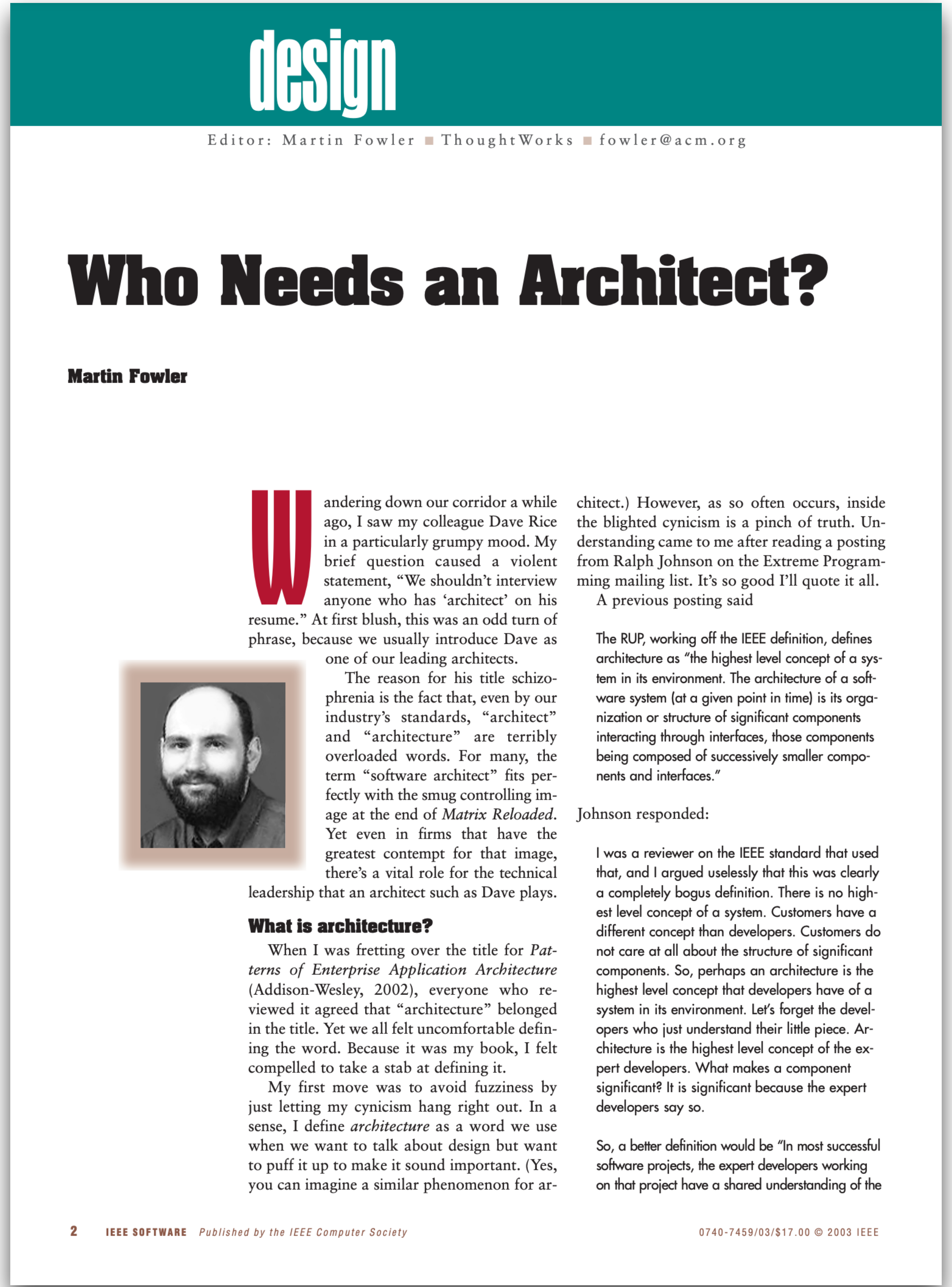
Fundamentals of FRONTEND ARCHITECTURE

WHAT IS SOFTWARE ARCHITECTURE?

SOFTWARE ARCHITECTURE IS ABOUT THE STRUCTURE OF A SYSTEM



Editor: Martin Fowler ■ ThoughtWorks ■ fowler@acm.org

Who Needs an Architect?

Martin Fowler

Wandering down our corridor a while ago, I saw my colleague Dave Rice in a particularly grumpy mood. My brief question caused a violent statement, "We shouldn't interview anyone who has 'architect' on his resume." At first blush, this was an odd turn of phrase, because we usually introduce Dave as one of our leading architects.

The reason for his title schizophrenia is the fact that, even by our industry's standards, "architect" and "architecture" are terribly overloaded words. For many, the term "software architect" fits perfectly with the smug controlling image at the end of *Matrix Reloaded*. Yet even in firms that have the greatest contempt for that image, there's a vital role for the technical leadership that an architect such as Dave plays.

What is architecture?

When I was fretting over the title for *Patterns of Enterprise Application Architecture* (Addison-Wesley, 2002), everyone who reviewed it agreed that "architecture" belonged in the title. Yet we all felt uncomfortable defining the word. Because it was my book, I felt compelled to take a stab at defining it.

My first move was to avoid fuzziness by just letting my cynicism hang right out. In a sense, I define *architecture* as a word we use when we want to talk about design but want to puff it up to make it sound important. (Yes, you can imagine a similar phenomenon for architect.) However, as so often occurs, inside the blighted cynicism is a pinch of truth. Understanding came to me after reading a posting from Ralph Johnson on the Extreme Programming mailing list. It's so good I'll quote it all.

A previous posting said

> The RUP, working off the IEEE definition, defines architecture as "the highest level concept of a system in its environment. The architecture of a software system (at a given point in time) is its organization or structure of significant components interacting through interfaces, those components being composed of successively smaller components and interfaces."

Johnson responded:

> I was a reviewer on the IEEE standard that used that, and I argued uselessly that this was clearly a completely bogus definition. There is no highest level concept of a system. Customers have a different concept than developers. Customers do not care at all about the structure of significant components. So, perhaps an architecture is the highest level concept that developers have of a system in its environment. Let's forget the developers who just understand their little piece. Architecture is the highest level concept of the expert developers. What makes a component significant? It is significant because the expert developers say so.
>
> So, a better definition would be "In most successful software projects, the expert developers working on that project have a shared understanding of the

0740-7459/03/$17.00 © 2003 IEEE

FRONTEND AT SCALE

“ARCHITECTURE IS THE DECISIONS YOU WISH YOU COULD GET RIGHT EARLY IN A PROJECT”

“ARCHITECTURE IS THE THINGS THAT PEOPLE PERCEIVE AS HARD TO CHANGE”

"ARCHITECTURE IS ABOUT THE IMPORTANT STUFF…

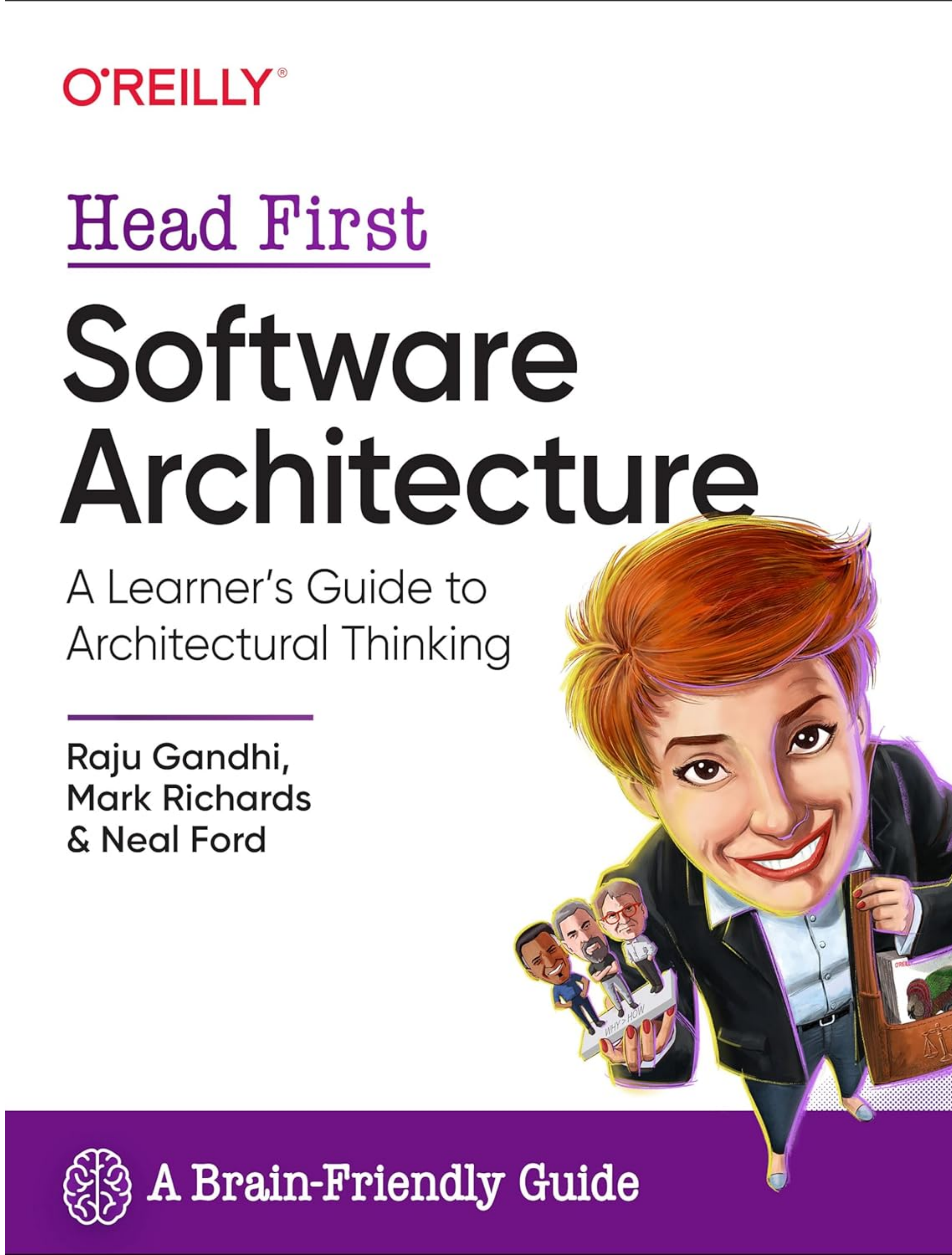
WHATEVER THAT IS."

“A system’s software architecture is the set of significant design decisions about how the software is organized to promote desired quality attributes and other properties.”

The Pragmatic Programmers
Design It!
From Programmer to Software Architect
Michael Keeling
edited by Susannah Pfalzer

THE FOUR DIMENSIONS OF ARCHITECTURE

- Architectural Style
- Architectural Characteristics
- Architectural Decisions
- Logical Components



O'REILLY

Head First

Software Architecture

A Learner's Guide to Architectural Thinking

Raju Gandhi, Mark Richards & Neal Ford

A Brain-Friendly Guide

CLASS

Archer

BACKGROUND

Origin

Personality

Backstory

STATS

STR	10
SPD	22
DEX	34
INT	16

SKILLS

ARCHITECTURAL STYLE

Microservices

BACKGROUND

Origin

Personality

Backstory

STATS

STR	10
SPD	22
DEX	34
INT	16

SKILLS

ARCHITECTURAL STYLE

Microservices

BACKGROUND

Origin

Personality

Backstory

ARCHITECTURAL CHARACTERISTICS

Performance	10
Scalability	22
Reliability	34
Agility	16

SKILLS

ARCHITECTURAL STYLE

Microservices

ARCHITECTURAL CHARACTERISTICS

Performance	10
Scalability	22
Reliability	34
Agility	16

ARCHITECTURAL DECISIONS

Structural

Guides

SKILLS

ARCHITECTURAL STYLE

Microservices

ARCHITECTURAL CHARACTERISTICS

Performance	10
Scalability	22
Reliability	34
Agility	16

ARCHITECTURAL DECISIONS

Structural

Guides

LOGICAL COMPONENTS

Modules

UI Components

Design System