



THE 21ST EDITION OF THE INTERNATIONAL CONFERENCE EUROPEAN INTEGRATION REALITIES AND PERSPECTIVES

From Code to User: Interdisciplinary Dimensions of Communication in Computer Science

Manuela-Andreea Ureche¹, Costel Iordache²

Abstract: Objectives: This paper examines the interdisciplinary dimensions of communication in computer science, arguing that communication quality is a critical success factor across all stages of software development. **Prior Work:** The study builds on foundational research in human-computer interaction, literate programming, data visualization theory, and recent advances in explainable artificial intelligence, integrating contributions from cognitive psychology, linguistics, and digital ethics. **Approach:** A theoretical and analytical approach was employed, synthesizing evidence from existing literature, empirical studies in software engineering, and current international ergonomics and web accessibility standards. **Results:** The paper identifies four distinct communication levels in software systems — intra-system, inter-developer, system-to-user, and data-mediated — and demonstrates that technical failures are frequently rooted in communication deficiencies rather than algorithmic shortcomings. **Implications:** The findings are relevant for computer science educators, software engineering practitioners, and academic researchers, suggesting that communication competencies should be formally integrated into IT curricula. **Value:** The paper proposes an original four-level conceptual model of communication in computer science, offering a structured interdisciplinary framework not previously consolidated in the literature.

Keywords: software usability; interface design; software docs; data visualization; HCI principles

1. Introduction

Communication represents one of the fundamental pillars of any organized human activity, and the field of computer science is no exception. From the earliest lines of assembly code to modern artificial intelligence applications, every stage of software development involves complex communicative acts: between developers, between systems, and between machines and humans.

Nevertheless, communication in computer science is frequently treated as a secondary concern, subordinate to technical aspects. This reductive perspective overlooks the reality that a software product, however technically performant, is ultimately ineffective if it fails to communicate efficiently with its end users. Recent studies in software engineering indicate that over 60% of IT project failures are

¹ Student, Danubius International University of Galati, Romania, Address: 3 Galați Blvd., Galați, Romania, Corresponding author: manuelaandreeaureche@univ-danubius.ro.

² Student, Danubius International University of Galati, Romania, Address: 3 Galati Blvd., Galați, Romania, E-mail: costel.iordachecostel@gmail.com



Copyright: © 2026 by the authors.
Open access publication under the terms and conditions of the
Creative Commons Attribution-NonCommercial (CC BY NC) license
(<https://creativecommons.org/licenses/by-nc/4.0/>)

attributable not to technical deficiencies, but to communication problems — unclear requirements, inadequate documentation, or confusing interfaces (Standish Group, 2023).

This paper adopts an interdisciplinary approach to communication in computer science, integrating contributions from linguistics, cognitive psychology, visual design, and digital ethics. The central objective is to demonstrate that improving communication quality at every stage of the software lifecycle constitutes a critical determinant of both technical and commercial success.

2. Related Work

The intersection of communication theory and computer science has attracted scholarly attention across multiple disciplines. Shannon and Weaver's (1949) foundational mathematical model of communication, originally developed for telecommunications, was later extended to human-computer interaction research. Watzlawick, Beavin and Jackson (1967) further enriched the theoretical framework by establishing that one cannot not communicate — a principle directly applicable to interface design.

In software engineering, the relationship between communication quality and project outcomes has been extensively documented. Brooks (1975) identified communication overhead as the primary cause of delays in large software projects. More recently, agile methodologies such as Scrum and Extreme Programming have placed interpersonal communication at the core of their development philosophy, prioritizing working collaboration over comprehensive documentation (Beck et al., 2001).

The field of Human-Computer Interaction (HCI) has produced a substantial body of research on system-to-user communication. Norman's (1988) concept of affordances — the perceived action possibilities of an interface element — remains a cornerstone of user-centered design. Nielsen's (1994) ten usability heuristics continue to guide interface evaluation and serve as a practical communication framework between designers and end users.

3. Communication Across the Software Lifecycle

3.1. From Requirements to Code: The Language of Specifications

The first critical stage in any software project is the transformation of user needs into technical specifications — an act of interdisciplinary translation. Natural client language must be converted into the formal language of software architecture. Modern methodologies such as Behavior-Driven Development (BDD) have formalized this transition through semi-natural syntaxes (Gherkin, Cucumber) that enable communication among business analysts, testers, and programmers without significant ambiguity.

Computational linguistics offers valuable tools at this stage: semantic analysis of requirements, detection of lexical ambiguities, and identification of logical contradictions can be partially automated through Natural Language Processing (NLP) techniques. Large language models such as BERT and GPT-4 are now employed in commercial requirements analysis tools, significantly reducing misinterpretation error rates.

3.2. Source Code as Communicative Text

Source code is not intended exclusively for computers — it is read, reviewed, and modified by humans. Knuth (1984) introduced the concept of literate programming, arguing that programs must be written primarily for human comprehension and only secondarily for machine execution. This vision profoundly influenced documentation culture in open-source development and gave rise to modern standards such as PEP 8 for Python, Google Style Guides, and the Clean Code principles articulated by Martin (2008).

Cognitive psychology contributes to understanding how programmers process code: eye-tracking studies have demonstrated that the visual structure of code — indentation, variable naming, and inline comments — directly affects comprehension speed and maintenance error rates. Descriptively named variables reduce code comprehension time by approximately 30% compared to abbreviated identifiers (Buse & Weimer, 2010).

4. The Interface: Meeting Point Between System and User

4.1. Human-Computer Interaction as a Communicative Act

The graphical user interface (GUI) constitutes the most visible layer of communication in computer science. From a communication theory perspective, a well-designed interface functions as a bidirectional communication channel: it transmits information about system state to the user and receives commands in return. The GOMS model (Goals, Operators, Methods, Selection Rules) proposed by Card, Moran and Newell (1983) remains a foundational framework for the cognitive analysis of user-system interactions.

User-Centered Design (UCD) principles, formalized in ISO 9241-210, mandate that interface development be grounded in validated user needs, established through qualitative and quantitative research. Techniques such as rapid prototyping, usability testing, and user journey mapping have transformed UI/UX design from an aesthetic activity into an evidence-based discipline.

4.2. Accessibility and Universal Communication

A frequently neglected dimension of communication in computer science is accessibility — the capacity of a system to communicate effectively with users with disabilities. The WCAG 2.1 standards establish precise criteria for visual contrast, keyboard navigation, screen reader compatibility, and textual alternatives for multimedia content. Compliance with these standards represents not only a legal obligation in many jurisdictions, but a fundamental ethical principle of inclusive communication.

Artificial intelligence introduces new possibilities in accessibility: automatic speech recognition, text synthesis, automated image description, and real-time translation significantly extend the communicative capabilities of individuals with visual, auditory, or motor impairments.

5. Data Visualization as Visual Language

In the era of big data, data visualization has emerged as one of the most significant forms of communication in computer science. Tufte (1983), in his seminal work on the visual display of quantitative information, defined the principles of a visual data language: clarity, precision, and efficiency in conveying quantitative information.

The psychology of visual perception — particularly Gestalt theory — underpins the design rules for charts and dashboards. Research in cognitive neuroscience has demonstrated that the brain processes visual information approximately 60,000 times faster than text, which accounts for the substantial impact of well-designed visualizations in analytical communication (MIT, 2019).

Modern libraries such as D3.js, Matplotlib, Plotly, and Tableau have democratized the creation of interactive visualizations, enabling data specialists to communicate complex insights to broad audiences without specialized technical backgrounds. This democratization of visual communication represents one of the major transformations of contemporary data science.

6. AI-Mediated Communication

6.1. Conversational Interfaces and Natural Language

The emergence of large language models (LLMs) — such as GPT-4, Claude, and Gemini — has revolutionized the paradigm of human-computer communication. Natural language interaction eliminates traditional technical barriers, enabling users without programming knowledge to access complex functionalities through conversation. This paradigmatic shift represents the most significant evolution in user interfaces since the introduction of the graphical interface in the 1980s.

Chatbots and virtual assistants integrated into mobile applications use NLP techniques to interpret user intent, extract relevant entities, and generate contextual responses. The quality of this communication depends equally on language model performance and on the design of prompts and conversational flows.

6.2. Ethical Dimensions of Algorithmic Communication

Algorithm-mediated communication raises unprecedented ethical challenges. Recommendation systems, medical chatbots, and educational assistants transmit messages with potentially significant impact on user decisions. Algorithmic transparency — the capacity of a system to explain its decisions in an intelligible manner — has become a fundamental principle of responsible AI, formalized in regulatory frameworks such as the EU AI Act (2024).

Explainable AI (XAI) techniques — SHAP, LIME, and Grad-CAM — aim to make ML system communication more transparent and credible for end users, reducing the black-box phenomenon and increasing trust in automated decisions.

7. An Integrated Conceptual Model of Communication in Computer Science

Based on the preceding analysis, this paper proposes a four-level model of communication in computer science, each level corresponding to a distinct communicative interface:

Level 1 — Intra-system communication: information transfer between software components (APIs, protocols, data formats). Relevant disciplines include software architecture and distributed systems engineering.

Level 2 — Inter-developer communication: code documentation, code reviews, technical specifications, and agile ceremonies. Relevant disciplines include technical linguistics and knowledge management.

Level 3 — System-to-user communication: graphical interfaces, error messages, notifications, and contextual help. Relevant disciplines include HCI, cognitive psychology, and UX design.

Level 4 — Data-mediated communication: reports, visualizations, dashboards, and conversational AI. Relevant disciplines include data science, visual communication, and digital ethics.

The model underscores that technical excellence at each level is necessary but insufficient — communication quality ultimately determines the perceived value of any information system. Each level builds upon the previous, forming a coherent chain from machine-readable instructions to meaningful human experience.

8. Analysis of Results

The interdisciplinary analysis conducted across five domains — software engineering, HCI, cognitive psychology, data science, and AI ethics — consistently confirms that communication failures represent a systemic risk in software development. The proposed four-level model provides a structured vocabulary for identifying where communication breakdowns occur and which disciplinary tools are most appropriate for remediation.

At Level 1, intra-system communication failures manifest as integration errors, data format mismatches, and API versioning conflicts. At Level 2, inter-developer failures produce maintenance debt, knowledge silos, and onboarding inefficiencies. At Level 3, system-to-user failures generate usability issues, user abandonment, and accessibility barriers. At Level 4, data-mediated failures result in misinterpreted analytics, algorithmic bias, and erosion of user trust.

The convergence of these findings across disciplines reinforces the central thesis: communication in computer science is not a peripheral soft skill but a core technical competency that must be systematically cultivated and evaluated throughout the software development lifecycle.

9. Conclusions

This paper has demonstrated that computer science, far from being a purely technical discipline, is traversed at all levels by profoundly communicative acts. From the semantics of a variable name to the design of an AI-powered conversational assistant, every decision made by a computer scientist carries communicative implications with direct impact on the usability, accessibility, and effectiveness of the systems produced.

The interdisciplinary approach adopted here — integrating linguistics, cognitive psychology, visual design, and ethics — is not an academic luxury but a practical necessity for 21st-century IT professionals. The training of future computer scientists must incorporate technical and interpersonal communication competencies as essential curriculum components, alongside algorithms and programming.

Future research directions include: empirical evaluation of documentation quality impact on software maintenance outcomes; development of communicability metrics for AI interfaces; and comparative studies of communication practices in globally distributed software development teams.

Acknowledgements

The author would like to thank Lect. univ. dr. Postolache Florin for academic supervision and guidance throughout the preparation of this paper.

References

- Beck, K., Beedle, M., van Bennekum, A., Cockburn, A., Cunningham, W., Fowler, M., ... Thomas, D. (2001). Manifesto for Agile Software Development. Agile Alliance.
- Brooks, F. P. (1975). *The Mythical Man-Month: Essays on Software Engineering*. Addison-Wesley.
- Buse, R. P. L., & Weimer, W. R. (2010). Learning a metric for code readability. *IEEE Transactions on Software Engineering*, 36(4), 546–558.
- Card, S. K., Moran, T. P., & Newell, A. (1983). *The Psychology of Human-Computer Interaction*. Lawrence Erlbaum Associates.
- European Parliament. (2024). Regulation (EU) 2024/1689 — Artificial Intelligence Act. Official Journal of the European Union.
- ISO 9241-210:2019. (2019). Ergonomics of human-system interaction — Part 210: Human-centred design for interactive systems. International Organization for Standardization.
- Knuth, D. E. (1984). Literate programming. *The Computer Journal*, 27(2), 97–111.
- Martin, R. C. (2008). *Clean Code: A Handbook of Agile Software Craftsmanship*. Prentice Hall.
- MIT News. (2019). In the blink of an eye: MIT neuroscientists find the brain can identify images seen for as little as 13 milliseconds. Massachusetts Institute of Technology.
- Nielsen, J. (1994). Heuristic evaluation. In J. Nielsen & R. L. Mack (Eds.), *Usability Inspection Methods* (pp. 25–62). Wiley.
- Norman, D. A. (1988). *The Design of Everyday Things*. Basic Books.
- Shannon, C. E., & Weaver, W. (1949). *The Mathematical Theory of Communication*. University of Illinois Press.
- Standish Group. (2023). *CHAOS Report 2023: Decision Latency Theory*. The Standish Group International.
- Tufte, E. R. (1983). *The Visual Display of Quantitative Information*. Graphics Press.
- W3C. (2018). *Web Content Accessibility Guidelines (WCAG) 2.1*. World Wide Web Consortium.
- Watzlawick, P., Beavin, J. H., & Jackson, D. D. (1967). *Pragmatics of Human Communication*. Norton.