

## Series Foreword

Software is deeply woven into contemporary life—economically, culturally, creatively, politically—in manners both obvious and nearly invisible. Yet while much is written about how software is used and the activities that it supports and shapes, thinking about software itself has remained largely technical for much of its history. Increasingly, however, artists, scientists, engineers, hackers, designers, and scholars in the humanities and social sciences are finding that for the questions they face, and the things they need to build, an expanded understanding of software is necessary. For such understanding they can call upon a strand of texts in the history of computing and new media, they can take part in the rich implicit culture of software, and they can also take part in the development of an emerging, fundamentally transdisciplinary, computational literacy. These provide the foundation for Software Studies.

Software Studies uses and develops cultural, theoretical, and practice-oriented approaches to make critical, historical, and experimental accounts of (and interventions via) the objects and processes of software. The field engages and contributes to the research of computer scientists, the work of software designers and engineers, and the creations of software artists. It tracks how software is substantially integrated into the processes of contemporary culture and society, reformulating processes, ideas, institutions, and cultural objects around their closeness to algorithmic and formal description and action. Software Studies proposes histories of computational cultures and works with the intellectual resources of computing to develop reflexive thinking about its entanglements and possibilities. It does this both in the scholarly modes of the humanities and social sciences and in the software creation/research modes of computer science, the arts, and design.

The Software Studies book series, published by the MIT Press, aims to publish the best new work in a critical and experimental field that is at once culturally and technically literate, reflecting the reality of today's software culture.

