Foreword

Computing is fast becoming ubiquitous and pervasive. It is *ubiquitous* because computing power and access to the Internet is being made available everywhere; it is *pervasive* because computing is being embedded in the very fabric of our environment. Xerox Corp. has recently coined the phrase “smart matter” to capture the idea of computations occurring within formerly passive objects and substances. For example, our houses, our furniture, and our clothes will contain computers that will enable our surroundings to adapt to our preferences and needs. New visions of interactivity portend that scientific, commercial, educational, and industrial enterprises will be linked, and human spheres previously untouched by computing and information technology, such as our personal, recreational, and community life, will be affected.

However, when there is information everywhere and all manner of things are interconnected, there arise the problems of information overload and misunderstandings. Dogbert, from Scott Adams’ Dilbert, describes the situation as “Information is gushing toward your brain like a fire hose aimed at a teacup.”

This book analyzes the problems of information overload and misunderstandings and provides a solution: a way for all of the different devices, components, and computers to understand each other, so that they will be able to work together effectively and efficiently. This is a powerful and important advance.

Dr. Vipul Kashyap and Professor Amit Sheth have long been leaders in the area of information system semantics and are widely known as two of the area’s most dedicated, productive, and insightful researchers. Together in this book they have crafted a coherent vision of the single most important element of a distributed heterogeneous information system: the information broker. Previously, Kashyap has been the architect of a broker-based multiagent system for cooperative information access. Sheth has been an innovator of metadata-based approaches to the integration of heterogeneous semantics for database systems and workflow systems. Together, their expertise is complementary and forms the unique perspective of this book.

The essential agent-based architecture that they describe and analyze is becoming canonical. Agents are used to represent users, resources, middleware, security, execution engines, ontologies, and brokering. As the technology advances, we can expect such specialized agents to be used as standardized building blocks for information systems. Two trends lend credence to such a prediction.

First, software systems in general are being constructed with larger components, such as ActiveX and JavaBeans, which are becoming closer to being agents themselves. They have more functionality than simple objects, respond to events autonomously, and, most importantly, respond to system builders at development-time, as well as to events at run-time.

Second, there is a move toward more cooperative information systems, in which the architecture itself plays an important role in the effectiveness of the system, as opposed to traditional software systems where effectiveness depends on the quality of the individual components. These are the architectures of standardized agents that Kashyap and Sheth elucidate. Architectures based on standardized agent types should be easier to develop, understand, and use. Perhaps most important of all, these architectures will make it easier for separately developed information systems to interoperate.

Among the reasons why agents are attractive, there are two main ones of interest here. One, agents enable the construction of modular systems from heterogeneous pieces, potentially created by any number of vendors. Two, the agents themselves embody diverse knowledge, reasoning approaches, and perspectives. This diversity is sometimes essential, because the agents represent people or business interests that have different goals and motivations. Diversity can sometimes be added in by design: it can make an agent system more robust by enabling a variety of viewpoints to be represented and exploited.

However, agents are typically complex pieces of software, so the question arises whether a set of different agents would unnecessarily add to a system's complexity. The more kinds of agents there are, the harder it might be to build and maintain them.

Fortunately, this turns out to be a false concern. The agents have to be diverse in content, e.g., knowledge, reasoning techniques, and interaction protocols, but not in the form in which that content is realized, e.g., the language or toolkit with which they are constructed. Problems arise through unnecessary heterogeneity in construction; the cost of necessary heterogeneity in content is more than recovered through the flexibility it offers.

In summary, the results in this book are applicable not only to the huge amount of information available globally over the World-Wide Web, but also to the diverse information soon to be available locally over household, automobile, and environment networks. I am excited by the possibilities for new applications and uses for information that are engendered by this book, as well as by the challenges that remain. This book provides a solid foundation for advances in distributed heterogeneous information systems.

Professor Michael N. Huhns
University of South Carolina