highly imperative means that distributed programs must be carefully crafted to avoid unexpected behavior due to unwanted copying of shared data.

[0015] In addition, neither Java nor Java/RMI permit an object to simultaneously span multiple heterogeneous machines. Each object is resident on exactly one machine at any given time. As a result, true concurrency on multiple machines within the encapsulation of a single object is impossible. Moreover, like a typical RPC system, communication among tasks using RMI is through copying. Thus, the semantics of a Java/RMI program may be quite different from a syntactically similar Java program.

Agent Languages

[0016] Besides RPC and Java, numerous proposals have been made for agent languages which allow computation and data to freely migrate within a network. Conceptually, an agent is an encapsulation of a computation (i.e. a task) and related data that is mobile (i.e. can freely move about within a distributed network of machines).

[0017] For example, Aglets, described for example in D.B. Lange et al., "Programming Mobile Agents in Java - With The Java Aglet API", IBM, URL = http://www.trl.ibm.com/aglets/agletbook, (1998), is a Java mobile agent system that uses the Java/RMI interface and a security manager to achieve a portable and secure agent system. However, Aglets do not permit an agent's state to simultaneously span multiple heterogeneous machines, and migration of an agent requires the entire agent to move from one machine to another.

[0018] Telescript and Odyssey are two other mobile agent languages. *See* J. White, "Mobile Agents White Paper," General Magic, URL = http://www.generalmagic.com/technology/techwhitepaper.html (1998); "Introduction to the Odyssey API," General Magic, URL = http://www.generalmagic.com/agents/odysseyIntro.pdf (1998). While both Telescript and Odyssey agents can migrate during execution, the state of such agents can only reside on a single machine at any given moment. Thus, Telescript and Odyssey agents do not allow distributed state: when an agent moves, it is necessary that its entire state moves along with it. This limitation significantly reduces functionality and efficiency. In the case of Odyssey, only the state as found in the heap can move--the state of the stack, program counter, and registers are all lost. Telescript imposes similar restrictions.

[0019] Other systems that support mobile computation are Agent Tcl (*see* D. Kotz et al., "AGENT TCL: Targeting the Needs of Mobile Computers," *IEEE Internet Computing*, Vol.1, No. 4, pp. 58-67 (1997)) and ARA (*see* H. Peine et al., "The Architecture of the ARA Platform for Mobile Agents," *Proceedings of the First international Workshop on Mobile Agents* (K. Rothermel et al., eds.), pp. 50-61 (1997)), whose base languages were originally Tcl but who have recently been provided Java support. Like Odyssey and Aglets, these systems also prohibit an agent from having distributed state, and provide no infrastructure by which an agent can transparently access data resident on another machine.

[0020] Obliq (*see* L. Cardelli, "A Language with Distributed Scope," *Proceedings of the 22nd ACM Symposium on Principles of Programming Languages*, pp. 286-298 (1995)) and Kali (*see* H. Cejtin et al., "Higher-Order Distributed Objects," *ACM Transactions on Programming Languages and Systems*, Vol. 17, No. 5, pp. 704-739 (1995)) are two other programming languages that permit code and data to migrate within a heterogeneous network. Obliq's sequential semantics is a delegation-based object system, whereas Kali is built on top of Scheme (*see* W. Clinger et al., eds. "Revised Report on the Algorithmic Language Scheme," *ACM Lisp Pointers*, Vol. 4, No. 3, pp. 1-55 (July 1991)), a higher-order lexically-scoped dialect of Lisp. Neither of these two systems have an explicit notion of agents, however. While Obliq supports transparent references, it does so by severely restricting the conditions under which objects may migrate. Moreover, Obliq does not provide a notion of a distributed address space such as an agent. Kali requires all operations on remote references to be explicitly performed. Like Obliq, Kali does not support an object-based encapsulation model. These limitations make Obliq and Kali ill-suited for large-scale distributed systems with mobile applications.

[0021] Accordingly, there remains a need for a distributed computing system which is easy to program and which: (1) provides an object-based encapsulation model, such as an agent, which allows the processes and state of the agent to be distributed over multiple potentially heterogeneous machines; (2) enables transparent access of data resident on another machine; and (3) allows easy and efficient process migration, in whole or in part, among distinct machines.

SUMMARY OF THE INVENTION

[0022] Generally speaking, in accordance with the invention, a distributed software system for use with a plurality of computer machines connected as a network is provided. The system may comprise a plurality of bases, each base providing a local address space and computer resources on one of a plurality of computer machines. At least one agent comprising a protection domain is provided, wherein the protection domain of the at least one agent resides on at least one of the plurality of bases. A plurality of objects are contained within the protection domain of the at least one agent, a first object residing on a first base of the plurality of bases and a second object residing on a second base of the plurality of bases. The first object on the first base may access the second object on the second base without knowledge of the physical address of the second object on the second base. Finally, at least one runtime system