Process Integrated Mechanism: A Novel Software For Space Missions

James Allen[1], Kenneth Ford[1], Robert Morris[2], Niranjan Suri[1]

[1]Institute for Human and Machine Cognition

[2]NASA Ames Research Center

Abstract—This paper will describe a novel software architecture for coordinating the behavior of distributed space exploration systems. Example applications range from configuration and control of small satellites, coordinated exploration with multiple automated rovers, and distributed actuation.

A Process Integrated Mechanism (PIM) is a novel software architecture for coordinating the behavior of multiple satellites, rovers, and any other computer based nodes that need to operate together to achieve a common objective. Current solutions to the problem of coordinating distributed complex systems are based on either centralized approaches or distributed approaches. Centralized approaches are simple to program and predict, but present a single point of failure and require robust communications to the node that is the central coordinator. Distributed approaches rely on notions of multi-agent systems or on emergent behavior using the notions of swarms, but are difficult to program, difficult to predict, and difficult to debug or adapt to a new environment. However, distributed approaches are robust since they do not present a single point of failure.

PIM is a novel architecture which leverages the advantages of centralized and distributed approaches into the same system. A PIM consists of a single Coordinating Process (CP) that is pseudo-simultaneously residing on and controlling multiple systems. This illusion is realized by rapidly and transparently moving a single CP between all the nodes in the system. The migration is managed automatically and transparently by the PIM runtime environment that is executing on each node. Transparent migration ensures that the CP is neither aware of the migration nor the identity of the specific node on which it might be executing at any given instant in time. Furthermore, the migration is fast enough so that the CP visits each node in sufficient time to meet the overall coordination needs of the system. Therefore, the PIM is the inverse of timesharing in computer systems. Timesharing executes multiple programs on a single CPU and switches between them quickly enough to provide the illusion that all the programs are running simultaneously. PIM executes a single program (the CP) on multiple CPUs and cycles it quickly enough to provide the illusion that the CP is simultaneously running on all nodes.

PIM also contributes to robustness by retaining the last executed copy on each of the nodes. If a node is lost or disconnected, it is temporarily removed from the set of nodes that comprise the PIM. If the CP happened to be resident on the lost node, then the PIM is restarted by recreating the CP from the last known state of the CP at the previous node.

A prototype of the PIM system has been designed and tested on a set of ActiveMedia Pioneer robots. The PIM runtime is based on the capabilities of the Aroma Java-compatible virtual machine, which provides the notion of strong mobility. The runtime executes CP developed using the Java programming language and initial performance results are promising. The PIM approach was also compared with a multi-agent approach, in terms of programming complexity, and the results are promising. A new prototype runtime system with additional optimizations and capabilities is currently under development.