

CPPvm - Parallel Programming in C++

Steffen Götz
Research and
Technology 3
Software Architecture

Topics

- What is CPPvm?
- Why should I use CPPvm?
- Data transfer
- Comparison
- Performance
- Statistics and applications
- What's next?

What is CPPvm?

CPPvm (C Plus Plus PVM) is a C++ class library for message passing built on top of PVM (Parallel Virtual Machine). CPPvm allows to:

- Combine a heterogeneous collection of computers
- Spawn and kill processes dynamically
- Detect failed processes and hosts
- Send/receive C++ objects
- Use distributed C++ objects
- Write your own message C++ classes
- Use standard template library (STL) classes
- Use semaphores
- Use mutual exclusion
- Use CPPvm together with existing PVM software

Why should I use CPPvm?

- It closes the gap between the design of object-oriented parallel programs in C++ and the underlying message passing possibilities of PVM
- Unique message passing features
- Easy-to-use
- Scales from simple examples to complex parallel programs
- Extensive documentation
- Available on many architectures

Comparison

Features	PVM 3.4.3	MPV 2.0	CPPvm 1.4.0	Pvpm 0.6.0	EasyPvm	Pvpm 2.1	COMPI 1.0.0
Process Handling							
Dynamic Processes							
Dynamic Nodes							
Global							
Global Notification Messages							
Process Synchronization							
Message Passing							
Global Data							
Local							
Semaphores							
Local Blocking							
Send Blocking							
Receive Blocking							
Receive Non-Blocking							
Receive Timeout							
Forward							
Group Broadcast							
Unicast							
Global Data							
Global							
Non-Local Objects							
Message Control							
Message Passing Control							
Global							
Communication Topologies							
Message Handler							
Parallel I/O							
Cache Coherency							
File							
Message Types							
Standard C Types							
Standard C++ Types							
Standard Template Library (STL)							
C++ Extensions							
User Defined Types							
Language Support							
C							
C++							
FORTRAN							
COBOL							
Fortran 90							
Fortran 95							
Fortran 2003							
Fortran 2008							
Fortran 2013							
Fortran 2018							
Fortran 2023							
Fortran 2024							
Fortran 2025							
Fortran 2026							
Fortran 2027							
Fortran 2028							
Fortran 2029							
Fortran 2030							
Fortran 2031							
Fortran 2032							
Fortran 2033							
Fortran 2034							
Fortran 2035							
Fortran 2036							
Fortran 2037							
Fortran 2038							
Fortran 2039							
Fortran 2040							
Fortran 2041							
Fortran 2042							
Fortran 2043							
Fortran 2044							
Fortran 2045							
Fortran 2046							
Fortran 2047							
Fortran 2048							
Fortran 2049							
Fortran 2050							
Fortran 2051							
Fortran 2052							
Fortran 2053							
Fortran 2054							
Fortran 2055							
Fortran 2056							
Fortran 2057							
Fortran 2058							
Fortran 2059							
Fortran 2060							
Fortran 2061							
Fortran 2062							
Fortran 2063							
Fortran 2064							
Fortran 2065							
Fortran 2066							
Fortran 2067							
Fortran 2068							
Fortran 2069							
Fortran 2070							
Fortran 2071							
Fortran 2072							
Fortran 2073							
Fortran 2074							
Fortran 2075							
Fortran 2076							
Fortran 2077							
Fortran 2078							
Fortran 2079							
Fortran 2080							
Fortran 2081							
Fortran 2082							
Fortran 2083							
Fortran 2084							
Fortran 2085							
Fortran 2086							
Fortran 2087							
Fortran 2088							
Fortran 2089							
Fortran 2090							
Fortran 2091							
Fortran 2092							
Fortran 2093							
Fortran 2094							
Fortran 2095							
Fortran 2096							
Fortran 2097							
Fortran 2098							
Fortran 2099							
Fortran 2100							

Data transfer



Explicit message passing:

- Send:
 - blocking
 - non-blocking
- Receive:
 - blocking
 - non-blocking
 - timeout
- Types:
 - CPPvm message classes
 - the standard C++ types bool, char, double, float, int and long as well as constants
 - the standard template library (STL) classes bitset, complex, deque, list, map, multimap, multiset, pair, priority_queue, queue, set, slist, stack, string, valarray, and vector

Distributed Objects:

- Read object from global database
- Write object to global database
- Types:
 - CPPvm message classes

User defined classes for:

- explicit message passing and
- distributed objects.

WWW statistics (4/99-8/01):
-> 11.000 requests