A Distributed Computing Support (DisCoS) Environment

Dr. Frederica Darema
DARPA

Outline
• Background: The technology gap
• A framework for new technology
• New Approaches
  • Programming /Compiling Technology
  • Application Composition Technology
  • System Analysis Technology
• Summary
Distributed Computing Support (DisCoS)

“Empowering Applications to exploit Future Distributed Heterogeneous Computing Systems”

Impact

Management enhanced with distributed applications

Example: Target Recognition
- Distributed execution
- Highly efficient execution
- Real time or faster than real time
- Improve accuracy of analysis

Improved process for application design, support and upgrade

Example: Pattern Recognition
- Reduce time to port applications: from months to ~hour
- Reduce cost to port applications: from $1M/port to $1K
- Enable rapid prototyping
- Reduce prototyping costs

Outline:
- Background: The technology gap
- Case example: DisCoS applied to MSTAR
- New approach
- Why now? Why the federal agencies?

DisCoS
- Distributed compiler and programming model
- Application assembly technology
- Computing system analysis technology

Enabling applications to efficiently execute across distributed, heterogeneous platforms

Enterprise/Scientific/Engineering Computing

Large, Complex, Heterogeneous Applications

Complex Distributed Adaptive Platforms

DisCoS: Distributed Systems Management
- Distributed Computing Platforms and Networks
- Other Runtime Services
- Distributed Computing Support (DisCoS)

Defense Applications

Darema -2
MSTAR
(Moving and Stationary Target Acquisition and Recognition)

SAR Image & Collateral Data
- DTED, DFAD
- Site Models
- EOSAT imagery

Target & Scene Model Database
(created off-line)

Indexing

Search

Clutter Database

Task Predict

ROI Hypothesis

Index Database
(created off-line)

Task Extract

Feature-to-Model Traceback

Match

Match Results
Form Associations Refine Pose & Score Analyze Mismatch

Score = 0.75

Shadow Obscuration?

Shadow

Grass

Road

Trees

H2O

Road

Trees

Grass
## Technology Gap for Distributed Applications

### Platform
- **Network of Workstations (NOW)**
- **Symmetric Multiprocessor (SMP)**
- **Cluster of SMPs**

### Programming Model
- **Message passing**
- **Static partition**
- **Shared queue**
- **Dynamic allocation of work**
- **Message-passing across SMPs**
- **Shared queue within SMP**

### Constraint
- **Inefficient load-balancing**
- **Application “re-write” required**
- **Application cannot be repartitioned dynamically when problem size or number of SMPs changes**

---

**Dynamic Analysis Situation**

**Launch Application(s)**

---

**Adaptable Systems Infrastructure**

---

**Distributed Computing Resources**

---

**Distributed Platform**

- MPP
- NOW
- SP
The Solution: DisCoS Technology

Dynamic Analysis Situation

Launch Application(s)

Dynamic Analysis Situation

Application Model

Application Program

Application Intermediate Representation

Dynamically Link & Execute

Distributed Computing Resources

DisCoS

Distributed Programming Model

Compiler Front-End

Compiler Back-End

Application Components

Architecture Models

Adaptable Systems Infrastructure

Distributed Platform

MPP
NOW
SP
Distributed Systems Software/Hardware
Architectural Framework

Distributed Defense Applications

(DisCoS) Distributed Computing Support

Collaboration Environments
Mission Support Environments
Security Dependability Survivability Services
Shared Storage Models
Other Services . . .

Distributed Systems Management

Distributed, Heterogeneous, Dynamic, Adaptive Computing Platforms and Networks

Memory Technology
CPU Technology
Device Technology
. . .

Application
API & Runtime Services
Global Management
Computing Engine
Components Technology
The Goal of this research is to create a System for Applications’ development and runtime support, comprised of:
- programming models and compilers which enable applications to dynamically map to the reconfigurable system infrastructure;
- software tools for dynamic selection of application components;
- analysis tools to enable delivering quality of service

The Impact: DisCoS will enable applications to:
exploit the changing features of the underlying distributed reconfigurable heterogeneous platforms, and satisfy dynamic demands

Techlogy Areas:

- **Application Programming System (APS):**
  distributed programming models and compilers

- **Application Composition System (ACS):**
  dynamic selection of distributed application components

- **Application Analysis System (AAS):**
  technology for performance engineered distributed applications

- **Validation, Integration and Demonstration:**
  validation, integration and demonstration of the technology
Technical Areas

• **Application Programming System** (distributed programming models and compilers)
  – distributed programming models for complex, distributed hardware platforms with complex memory structure and be adaptable to changes in the underlying platforms
  – interfaces that allow applications to specify performance related parameters to enable applications to achieve quality of service
  – compilers that interface with models of the underlying distributed hardware and software platforms to allow retargeting and optimizing application mappings on such complex systems

• **Application Composition System** (dynamic selection of distributed application components)
  – technology for building knowledge-based systems allowing automatic selection of solution methods allowing applications to adapt to changes in the underlying platforms or to changes in the application problem
  – application interfaces and methods for problem specification and extracting content information, standards of interfaces, data representation and data exchange, and standard high-level and low-level libraries
  – interfaces to debugging tools and performance models

• **Application Analysis System** (technology for performance engineered distributed applications)
  – modeling languages and models for application and system description
  – multi-resolution levels of data abstraction for interoperability of performance models of different levels of abstraction
  – methods and tools for measurement and instrumentation
  (initial efforts for this technology are supported under the as a result of the DARPA BAA 97-12, issued by F. Darema)

• **Validation, Integration and Demonstrations** (validation, integration and demo of the technology)
  – validation of key technologies developed under each of the tasks above
  – identify integrator to integrate the technologies developed above
  – demonstration of the ability of these technologies for design and runtime support of key applications executing under dynamically changing conditions (examples: Target/Pattern Recognition)
The present models poorly serve applications that:

- Require dynamic task scheduling and resource allocation
  - Tree-based algorithms (database searches)
  - DD applications, e.g. weather simulations using adaptive-mesh refinement
- Need task independence in the presence of multiple levels of memory hierarchy
- Must run on distributed heterogeneous platforms

**DisCoS will overcome the limitations with a new model and compiler technology:**

- Extend SPMD, compiler directives, negotiation with Operating System
  - Dynamically chunk the queue (ROI example)
- Develop a hybrid model combining features of existing models
- Create a new model allowing multiple levels of concurrency and data distribution
Present approaches for application software reuse and composition:
- Libraries of application kernels
- Libraries for specific models of memory hierarchy
- Problem Solving Environments enable “wiring together” specific application

Problems not addressed by existing technology:
- Find and select compatible software components to build applications for heterogeneous platforms
- Bridge different data models used by components
- Build applications by dynamically composing independently-developed components

New technology:
“user interface + libraries + knowledge-base“

- Straightforward extensions of existing technology:
  - Develop knowledge-based systems of components for specific defense applications
  - Populate the knowledge data-base of components for specific platforms
- Medium Risk Research Agenda
  - Develop efficient data exchange mechanisms between different data representations
  - Use data-mining to extract performance knowledge of specific application components
  - Develop general interface mechanisms for selecting suitable components
- High Risk Research Agenda:
  - Automatic generation of application beginning from high-level specifications (e.g. text, equations)
Distributed Systems Software/Hardware
Architectural Framework

(DiSCoS)
Distributed Computing Support

Visualization
Scalable I/O
Data Management
Archiving/Retrieval Services
Collaboration Environments
Authentication/Authorization
Dependability Services
Other Services . . .

Distributed Systems Management
Distributed, Heterogeneous, Dynamic, Adaptive Computing Platforms and Networks

Memory Technology
CPU Technology
Device Technology

Application
API
&
Runtime Services
Global Management
Computing Engine
Components Technology

Performance Engineered Design Technology
Distributed Systems Software/Hardware
Architectural Framework

Application Models
... 
IO / File Models
OS Scheduler Models
Architecture / Network Models
Memory Models

Distributed Applications

(DiSCoS)
Distributed Computing Support

Visualization
Collaboration Environments
Scalable I/O Data Management
Archiving/Retrieval Services
Authentication/Authorization Dependability Services
Other Services ...

Distributed Systems Management

Distributed, Heterogeneous, Dynamic, Adaptive Computing Platforms and Networks

Memory Technology
CPU Technology
Device Technology
...

Application
API
&
Runtime Services

Global Management
Computing Engine
Components Technology
Application Analysis System
Challenges and Approaches

Present methods and tools for performance analysis

- Modeling (queuing and analytical models)
- Simulation tools
  - architecture, network, cache, and I/O simulators
  - trace-driven, execution-driven simulations
- Performance data generation and collection
  - software assists (user directives, libraries)
  - hardware monitors
- On-line analysis and post analysis; Visualization

Problems with present technology

- Existing performance methods and tools study isolated system components
- Interaction of design features across different system layers not well understood
- No means to exploit design information at one level for another level (compiler-architecture for optimization of data mapping, or task scheduling)
- Dynamically-changing heterogeneous systems are even harder to analyze
- Current technology cannot be used to predict performance of future systems and allow reduction of prototyping effort

DisCoS approach for performance analysis:

- **Low Risk Research Agenda:**
  - Enable optimizations via application directives to the compiler
  - Develop simple parametric models of the application and underlying platform
- **Mid-Risk Research Agenda:**
  - Use parametric application models with system software and hardware models for optimizing task scheduling and partitioning by the compiler
- **High Risk Research Agenda:**
  - Develop performance frameworks with multi-resolution, integrable models across all levels of the system hierarchy, for more accurate compiler optimizations and mapping, and with capability to predict performance of the computing system, as a whole and across levels
Validation, Integration and Demos
Technical Approach

- Select defense applications; candidates: Large industrial applications
- Identify modules in these applications for validation of DisCoS components, like programming models, compiler techniques
  - Use these modules for demo-ing compiler capability to map these modules across platforms.
  - Test compiler ability to use resource information and architecture models
- Integrate the DisCoS components into the DisCoS system
- Demo use of DisCoS on development and runtime support of these application modules on distributed, heterogeneous platforms
DisCoS Roadmap

Application Programming System
- Distributed programming models
- Application performance Interfaces
- Compilers optimizing mappings on complex systems

Application Composition System
- Automatic selection of solution methods
- Interfaces, data representation & exchange
- Debugging tools

Application Analysis System
- Application/system multi-resolution models
- Modeling languages
- Measurement and instrumentation

DisCoS... Providing enhanced capabilities for applications
Role of the Federal Agencies

• The development of this technology will need initiative from the Federal Agencies
• Effort analogous to one that pushed the frontiers for parallel and scalable computing
• Build on cross-agency co-ordination
• Hard to do given the shrinking budgets and mission oriented programs
• … but…
Why Not Industry

- Desktop is the driver for commercial software, and industry focuses on producing flexible software for the low-end.
- "Commercial/Enterprise Computing" also poses requirements for more flexible and adaptable, reconfigurable interoperating systems and applications.

..... BUT.....

- Industry focused on the short term returns, rather than investing on research for the enabling technology.
- Moreover, industry addresses the problem by providing services for:
  - application porting
  - application integration
- .... and making LOTS of $s!!!

*Services* is the fastest growing component of computer vendors’ business.

- Industry however has history of adapting and productizing research technology which has demonstrated success.
DisCoS Benefits to Future Computing

**The Future Computing Environment:**

- Need for time-sensitive results
- Large applications, not computationally uniform
- Applications spanning multiple machines, dynamic resource needs
- Hierarchical distributed architecture computing platforms
- Resources with time varying availability

**The DisCoS system:**

- Creates ability for complex defense applications to be distributed, adaptable to reconfigurable platforms, and achieve quality of service
- Decreases time to rebuild distributed applications by two orders of magnitude (from months to ~hours)
- Reduces prototyping time and cost of large distributed applications (from many months to days, and from many $Ms to few $Ks)
- Preserves application investment over time and as hardware platforms change