Designing and Building
System Software (& Applications) to eliminate the barriers to
Petaflops Computing,
and beyond...

Dr. Frederica Darema
NSF
**MSTAR** (DARPA)
(Moving and Stationary Target Acquisition and Recognition)

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

**SAR Image & Collateral Data**

- DTED, DFAD
- Site Models
- EOSAT imagery

**Index Database**
(created off-line)

**Indexing**

**Search Tree**

**ROI Hypothesis**

- BMP-2
- Local Scene Map

**Task Predict**

**Predict**

**Feature-to-Model Traceback**

**ROI Hypothesis**

- BMP-2
- Local Scene Map

**Task Extract**

**Extract**

**Match Results**

- Form Associations
- Refine Pose & Score
- Analyze Mismatch

**Shadow Obscuration**

**Score = 0.75**

**Match**

- Score = 0.75

**Index Database**
(created off-line)

**Target & Scene Model Database**
(created off-line)

**Search**

**Ground Clutter**

**Tree Clutter**

**Statistical Model**

- CAD
- Semantic Tree
Application Directions

Past
- Mostly monolithic
- Mostly one programming language
- Computation Intensive
- Batch
- Hours/days

Present / Future
- Multi-Modular
- Multi-Language
- Multiple Developers
- Multi-source Data
- Computation Intensive
- Data Intensive
- Real Time
- Few Minutes/hours
- Visualization (real time)
- Interactive Steering

Change at faster pace
Platform Directions

**Past**
- Vector Processors
- SIMD MPPs

**Present**
- Distributed Memory MPs
- Shared Memory MPs

**Future**
- Distributed Platforms, Heterogeneous Computers and Networks
  - Heterogeneity
    - architecture
      - compute & network
    - node power
      - supernodes, PCs
  - Latencies
    - variable (internode, intranode)
  - Bandwidths
    - different for different links
    - different based on traffic
  - Multiple -levels of mem/mets/bw/lat

Past

Present

Future

GiBs

Grids

Petaflops Platform (Grid-in-a-Box)
Challenges

- Application Complexity (including DIGs)
- Architecture complexity \([\text{Grids and GiBs}]\)
  - Heterogeneity
  - Dynamic/Adaptive environments
- Quality of service:
  - Efficiency
  - Fault-tolerance
  - High confidence
**Manufacturing**: “Waterfall Model”

**Computer world**: “Throw over the fence”
Past Design Methodology for computing systems

- Many ingenious solutions, but also many ad hoc solutions, ...
- Modeling and simulations for individual components
- Isolated islands of methods and tools
- No notion of modeling system performance
  - no path from component to system behavior
  - no path from analysis to prediction
- Performance measurements
  - for workload characterization
  - to detect bottlenecks/bugs

... mostly an afterthought

Systems too complex to follow the same practices today
A Methodology for Performance Engineering Computing Systems
(Applications/System Software/Hardware-Architecture)
Computing Systems Software/Hardware Architectural Framework

Performance Engineered Design Technology

Distributed Applications

Languages
Compilers
Libraries
Tools

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

OS

Distributed, Heterogeneous, Dynamic, Adaptive Computing Platforms and Networks

Memory Technology
CPU Technology
Device Technology

Application
API & Runtime Services
Global Management
Computing Engine
Components Technology
Computing Systems Software/Hardware Architectural Framework

Application Models

IO / File Models

OS Scheduler Models

Architecture / Network Models

Memory Models

Distributed Applications

Languages

Compilers

Libraries

Tools

Visualization

Scalable I/O

Data Management

Archiving/Retrieval

Services

Authentication/Authorization

Dependability

Services

Other Services . . .

Operating System

Distributed, Heterogeneous, Dynamic, Adaptive Computing Platforms and Networks

Memory Technology

CPU Technology

Device Technology

. . .

Components Technology

Application

Runtime Services

Global Management

Computing Engine

Darema 6/28/99
Multiple views of the system
The applications’ view
Multiple views of the system
The Operating Systems’ view

Distributed Applications
- 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

Languages
- Compilers
- Libraries
- Tools

IO / File Models

OS Scheduler Models

Architecture / Network Models

Memory Models

Darema 13 6/28/99
New Performance Methodology

- **System approach**
  - consider the system in terms of its architectural layers
  - relate effects of different layers

- **Performance Frameworks**
  - combine modeling methods and tools in “plug-and-play” fashion
  - multiple views of the system

- **Multi-level and multi-modal approaches**
  - combine different methods of describing components and layers
  - ability to describe the system in multiple levels of detail (characteristics and time-scales)
Some efforts towards this direction

- DARPA BAA, Jan 1997
  - several projects started
    - Jim Browne and co-PI’s
    - Dan Reed and co-PI’s
    - Joel Saltz and co-PI’s
    - Graham Nudd and co-PI’s
    - Fran Berman and co-PI’s
  - + some SBIRs
- UMD sponsored workshop, Annapolis, Aug 19-20, 1998
- SC98 Performance Panel, Nov. 13, 1998 / Friday 8:30-10am
Future design practices:

“Integrated Engineering”

- Requirements
- Design
- Development/Manufacturing
- Operations & Management

Cost & Complexity of Software & Hardware

Need Performance Technology for

Design, Management, Control
An Integrated Environment for Application Development and Run-time Support

(New Compiling and Application Composition Technology)

*NGS/CADSS
The Solution: CADSS Technology

Dynamic Analysis Situation

Launch Application(s)

Application Model

Application Program

Application Intermediate Representation

Dynamically Link & Execute

CADSS

Distributed Programming Model

Compiler Front-End

Compiler Back-End

Application Components

Architecture Models

Distributed Computing Resources

Distributed Platform

Adaptable Systems Infrastructure (Grids or GiBs)

MPP

NOW

SP
Technology Innovations Needed

- **Application Support System** (distributed programming models and compilers)
  - programming models for complex, distributed hardware platforms with complex memory structure
  - compilers that interface with the system resource managers to request and determine resources, as well as interfacing with models of the underlying distributed hardware and software platforms to allow retargeting and optimizing application mappings on such complex systems
  - interfaces that allow applications to specify performance related parameters to enable applications to achieve quality of service

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

- **Application Analysis System** (technology for performance engineered applications)
  - modeling languages and models for application and system description
  - interoperability of performance models of different levels of abstraction
  - methods and tools for measurement and instrumentation

- **Validation, Integration and Demonstrations**
  - validation of key technologies developed under each of the thrusts above
  - integration of the technologies developed above
  - demonstration of the ability of these technologies for design and runtime support of key applications executing under dynamically changing conditions

Darema
Validation, Integration and Demos

Technical Approach

- Multidisciplinary research - Apps&Algs/SysSw/Arch&Hw
- Select challenging applications; large production-class applications
- Use these applications (or modules) for validation of CADSS 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 CADSS components into a CADSS system
- Demo use of CADSS on development and runtime support of these application modules on distributed, heterogeneous platforms
CADSS Technology 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

CADSS... Providing enhanced capabilities for applications
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

Darema
Role of the Federal Agencies

- The development of this technology will need initiative from the Federal Agencies
- Initiative analogous to one that pushed the frontiers for parallel and scalable computing
- Build on cross-agency co-ordination
- Today the budget better that it has been for years!
Summary Qs to Panelists - and As

• What responsibilities distinguish Peta-SysSw from present
  – A: MANY
    • complexity, heterogeneity, multi-levels of mem/nets, dynamic/adaptivity
    • need to position and convince that simulation is extremely powerful methodology - not only for scientific but also commercial (like engineering/mfg/finance/crisis.mgmt)

• Need paradigm shift in relation between Apps/SysSw/Arch?
  – A: YES.
    • Need to take system view. Performance Engineering Technology
    • Progress needed at all levels Apps+App.Algo/SysSw/Hw+Arch
    • New App. Support and Composition

• Major obstacles for Peta-SysSw? New areas?
  – A: Not obstacles. Challenges and Opportunities
    • Performance Engineering Technology important, App.Support and Composition

• Special Reqs on Hw
  – A: MANY, like:
    • High Bw, Low latency at all levels, address contention
    • monitoring (on-line performance data)
BACK-UP SLIDES
Outline

- Need for new Software Technology
  - background / motivation
  - examples

- Technology for performance engineered computing systems
  - analysis and prediction
  - multilevel methods, performance frameworks
  - some new research efforts

- New software architecture for application development and run-time support
  - new compiling system technology
  - application composition technology

New NSF program: New Generation Software

recently announced...
Systems Software/Hardware Architectural Framework

- **Applications/Users**
  - Languages
  - Compilers
  - Libraries
  - Tools
  - Visualization
  - Scalable I/O Data Management
  - Archiving/Retrieval Services
  - Collaboration Environments
  - Authentication/Authorization Dependability Services
  - Other Services . . .

- **Operating System**
  - Distributed, Heterogeneous, Dynamic Computing Platforms and Networks

- **Application**

- **Runtime Services**

- **Global Management**

- **Computing Engine**

- **Components Technology**
  - Memory Technology
  - CPU Technology
  - Device Technology
  - . . .
Distributed Systems Software/Hardware Architectural Framework

**Distributed Applications**

- Collaboration Environments
- Mission Support Environments
- Security Dependability Survivability Services
- Visualization
- Other Services . . .

**Distributed Systems Management**

- Distributed, Heterogeneous, Dynamic, Adaptive Computing Platforms and Networks

**Application**

- Runtime Services
- Global Management
- Computing Engine
- Components Technology

**Other Services**

- (CADSS) Complex Application Support System
- Mission Support Environments
- Visualization
- Other Services . . .

**Memory Technology**

**CPU Technology**

**Device Technology**

Darema
Push for new design methodology

- Design for performance from the outset
- Develop methods that describe components of the system, but also the system as a total
- Need the ability to describe, analyze and predict the behavior of components and of the system
- Analysis and prediction due to changes in the application, system software, hardware
- Challenges:
  - complexity of the system
  - cannot describe all components and all levels with the same detail
  - full-blown, detailed simulation too expensive, time consuming, impractical, intractable
Performance Technology Innovations Needed

- Modeling languages for modeling and specification of performance attributes for such components and layers
  - applications
  - system software
  - hardware
- Methods of modeling and simulation at multiple levels of detail and abstraction
- Models and simulators embodying these methods
- Ability to combine such models and simulation tools describing each layer or component at different levels of detail and different time-scales
- Development of interfaces so that models and simulators with different resolution levels can be combined into the performance frameworks
Performance Technology Innovations Needed (cont’d)

- Measurement methods and tools
  - including instrumentation methods to address heterogeneity (in systems and time-scales)
- Methods for combining and displaying such measurements
- Storage and retrieval of such measurements and other performance data
- Integration of these technologies into performance frameworks
CADSS: Technology for an integrated feedback and control compiling system

Dynamic Analysis Situation

Launch Application(s)

Dynamically Link & Execute

Application Intermediate Representation

Application Program

Application Model

Distributed Programming Model

Compiler Front-End

Compiler Back-End

Application Components

Architecture Models

Distributed Computing Resources

Distributed Platform

MPP

NOW

SP

Darema33

6/28/99

Dynamic Analysis

Situation

Launch Application(s)

Dynamically Link & Execute

Application Intermediate Representation

Application Program

Application Model

Distributed Programming Model

Compiler Front-End

Compiler Back-End

Application Components

Architecture Models

Distributed Computing Resources

Distributed Platform

MPP

NOW

SP

6/28/99 Darema33

Dynamically Link & Execute

CADSS: Technology for an integrated feedback and control compiling system

Adaptable Computing Systems (Grids or Grids)

Distributed Platform

MPP

NOW

SP

6/28/99 Darema33

Dynamically Link & Execute

CADSS: Technology for an integrated feedback and control compiling system

Adaptable Computing Systems (Grids or Grids)
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

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

- 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
**Application Composition System**

**Challenges and Approaches**

**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
- Develop algorithms tolerant to differences in different platforms
- 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
- Mid-???:
  - 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:
  - Automatic generation of application beginning from high-level specifications (e.g. text, equations)
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

new technology for performance analysis:

- Low Risk:
  - Enable optimizations via application directives to the compiler
  - Develop simple parametric models of the application and underlying platform
- Mid-Risk:
  - Use parametric application models with system software and hardware models for optimizing task scheduling and partitioning by the compiler
- High Risk:
  - Develop performance frameworks with multi-resolution, integrable models across all levels of the system hierarchy, for more accurate compiler optimizations and mapping
The President’s Information Technology Advisory Committee

Some findings:

“We cannot safely extend what we currently know to more complex systems”

“Learning how to build large-scale, highly reliable and secure systems requires research”

Recommendation:

“Increase funding in research and development of core software... “
Deadlines & Other Info

• Deadlines:
  – Dec 15, 1998 - letter of intent (+ list of reviewers)
  – Jan 12, 1999 - proposal

• Proposal $-size:
  – $200K - $1M per year
  – 3-year awards

• Tentative number of awards:
  – 10-12 of the higher $-range
  – 20-30 of lower $-range

• Reviews:
  – Panels + ad-hoc reviews
Additional Program Info

- NSF web pages / CISE and Darema
- NGS announcement Forums:
  - BOF at SC98 - Nov 10, 5:30-7:00pm
  - NCSA, Nov 16, 1:30-3:00pm (CST)
  - SDSC, Nov 20, 10:30am-12:00(PST)
- Related Panels:
  - SC98 - Panel on Performance Technology,
    Nov 13, 8:30-10:00am
Distributed Computing Support (CADSS)

“Empowering Applications to exploit Future Distributed Heterogeneous Computing Systems” (Grids and GiBs)

Enterprise/Scientific/Engineering Computing

Large, Complex, Heterogeneous Applications

Complex Distributed Adaptive Platforms

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

Enabling applications to efficiently execute across distributed, heterogeneous, and petaflops platforms

Defense Applications

Complex Application Support (CADSS) Other Runtime Services

Distributed Systems Management

Distributed Computing Platforms and Networks

Impact

Enabling complex 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

Darema
## Technology Gap
### Example case: Distributed Application

<table>
<thead>
<tr>
<th>Platform</th>
<th>Programming Model</th>
<th>Constraint</th>
</tr>
</thead>
<tbody>
<tr>
<td>Network of Workstations (NOW)</td>
<td>Message passing • Static partition</td>
<td>Inefficient load-balancing</td>
</tr>
<tr>
<td>Symmetric Multiprocessor (SMP)</td>
<td>Shared queue • Dynamic allocation of work</td>
<td>Application “re-write” required</td>
</tr>
<tr>
<td>Cluster of SMPs</td>
<td>Message-passing across SMPs • Shared queue within SMP</td>
<td>Application cannot be repartitioned dynamically when problem size or number of SMPs changes</td>
</tr>
</tbody>
</table>

**Launch Application(s)**

**Dynamic Analysis Situation**

**Adaptable Systems Infrastructure**

**Darema**

**Distributed Computing Resources**

**Distributed Platform**

- MPP
- NOW
- SP