Middleware Services for Sensors Systems in Dynamic Data-driven Oil Applications

Manish Parashar
The Applied Software Systems Laboratory
ECE/CAIP, Rutgers University
http://www.caip.rutgers.edu/TASSL

(Ack: NSF, DoE)
Outline

- Pervasive Grid Environments and Data-driven Management of Subsurface Geosystems
  - The Instrumented Oil Field

- Project AutoMate and Meteor: Middleware Services for Sensors Systems

- Deployments and Evaluations

- Concluding Remarks
Pervasive Computational Ecosystems and Dynamic Data Driven Applications

Experts query, configure resources

Computers, Storage, Instruments, ...

Resources discovered, negotiated, co-allocated on-the-fly. Components deployed

Experts interact and collaborate using ubiquitous and pervasive portals

Experts monitor/interact with/interrogate/steer models (“what if” scenarios,…). Application notifies experts of interesting events.

Component(s) write into the archive

Experts mine archive, match real-time data with history

Real-time data assimilation/injection (sensors, instruments, experiments, data archives).

Automated mining & matching

Resources

Applications & Services

Data Archive & Sensors

Model A

Model B

Data Archives

Sensors, Non-Traditional Data Sources

Components dynamically composed. “WebServices” discovered & invoked.
Pervasive Grid Environments – Unprecedented Opportunities

- Pervasive Grids Environments
  - Seamless, secure, on-demand access to and aggregation of, geographically distributed computing, communication and information resources
    - Computers, networks, data archives, instruments, observatories, experiments, sensors/actuators, ambient information, etc.
  - Context, content, capability, capacity awareness
  - Ubiquity and mobility

- Knowledge-based, information/data-driven, context/content-aware computationally intensive, pervasive applications
  - Symbiotically and opportunistically combine services/computations, real-time information, experiments, observations, and to manage, control, predict, adapt, optimize, …
    - Crisis management, monitor and predict natural phenomenon, monitor and manage engineering systems, optimize business processes

- A new paradigm?
  - seamless access
    - resources, services, data, information, expertise, …
  - seamless aggregation
  - seamless (opportunistic) interactions/couplings
Data-driven Management of Subsurface Geosystems: The Instrumented Oil Field (with UT-CSM, UT-IG, OSU, UMD, ANL)

Detect and track changes in data during production.
Invert data for reservoir properties.
Detect and track reservoir changes.
Assimilate data & reservoir properties into the evolving reservoir model.
Use simulation and optimization to guide future production.
Dynamic, Data Driven Reservoir Management

Dynamic Decision System

- Optimize
  - Economic revenue
  - Environmental hazard
  - ...
- Based on the present **subsurface knowledge** and **numerical model**
  - Update knowledge of model
  - Improve numerical model

Dynamic Data-Driven Assimilation

- Improve knowledge of subsurface to reduce uncertainty
  - Acquire remote sensing data
  - Plan optimal data acquisition
- Subsurface characterization
  - Data assimilation
- Experimental design
  - START

Management decision
Vision: Diverse Geosystems – Similar Solutions
Management of the Ruby Gulch Waste Repository (with UT-CSM, INL, OU)

- Ruby Gulch Waste Repository/Gilt Edge Mine, South Dakota
  - ~ 20 million cubic yard of waste rock
  - AMD (acid mine drainage) impacting drinking water supplies

- Monitoring System
  - Multi electrode resistivity system (523)
    - One data point every 2.4 seconds from any 4 electrodes
  - Temperature & Moisture sensors in four wells

- Flowmeter at bottom of dump
- Weather-station
- Manually sampled chemical/air ports in wells
- Approx 40K measurements/day

Pervasive Grid Applications – Unprecedented Challenges: Uncertainty

• System Uncertainty
  – Very large scales
  – Ad hoc structures/behaviors
    • p2p, hierarchical, etc, architectures
  – Dynamic
    • entities join, leave, move, change behavior
  – Heterogeneous
    • capability, connectivity, reliability, guarantees, QoS
  – Lack of guarantees
    • components, communication
  – Lack of common/complete knowledge
    • number, type, location, availability, connectivity, protocols, semantics, etc.

• Information Uncertainty
  – Availability, resolution, quality of information
  – Devices capability, operation, calibration
  – Trust in data, data models
  – Semantics

• Application Uncertainty
  – Dynamic behaviors
    • space-time adaptivity
  – Dynamic and complex couplings
    • multi-physics, multi-model, multi-resolution, ....
  – Dynamic and complex (ad hoc, opportunistic) interactions
  – Software/systems engineering issues
    • Emergent rather than by design
Pervasive Grid Computing – Research Issues, Opportunities

• Programming systems/models for data integration and runtime self-management
  – components and compositions capable of adapting behavior, interactions and information
  – correctness, consistency, performance, quality-of-service constraints

• Content-based asynchronous and decentralized discovery and access services
  – semantics, metadata definition, indexing, querying, notification

• Data management mechanisms for data acquisition and transport with real time, space and data quality constraints
  – high data volumes/rates, heterogeneous data qualities, sources
  – in-network aggregation, integration, assimilation, caching

• Runtime execution services that guarantee correct, reliable execution with predictable and controllable response time
  – data assimilation, injection, adaptation

• Security, trust, access control, data provenance, audit trails, accounting
Project AutoMate: Enabling Autonomic Applications

- Conceptual models and implementation architectures
  - programming systems based on popular programming models
    - object, component and service based prototypes
  - content-based coordination and messaging middleware
  - amorphous and emergent overlays
- http://automate.rutgers.edu
Project Meteor: Middleware Stack

- A self-organizing tiered overlay network
- Content-based routing engine (Squid)
  - Flexible content-based routing and querying with guarantees and bounded costs
  - Decentralized information discovery
- Associative Rendezvous Messaging
  - Symmetric post programming primitive
  - Content-based decoupled interaction with programmable reactive behavior
  - Decentralized content-based in-network aggregation (assimilation)
- Service-based API (WS)
Project Meteor: Associative Rendezvous

- Content-based decoupled interaction with programmable reactive behaviors
  - **Messages** - *(header, action, data)*
    - Symmetric post primitive: does not differentiating between interest/data
  - **Associative selection**
    - *match between interest and data*
  - **Reactive behavior**
    - *Execute action field upon matching*
- Decentralized in-network aggregation
  - *Tries for back-propagating and aggregating matching data items*
- Supports WS Notification standard
Implementation/Deployment Overview

- Current implementation builds on JXTA
  - SquidTON, Squid, Comet and Meteor layers are implemented as event-driven JXTA services

- Deployments include
  - Campus Grid @ Rutgers
  - ORBIT wireless testbed (400 nodes)
    - 400 802.11 radio nodes with 802.11 wireless connections
  - PlanetLab wide-area testbed
    - At least one node selected from each continent
ORB1T Testbed for Evaluation of Next Generation Wireless Networks: Phase 1: Indoor Grid
ORBIT Phase 2: Field Trial

- Requires ruggedized outdoor ready equipment (suggesting of the shelf technologies)
- Standard nodes used in dual role of mobile AP/mobile nodes deployed on busses
- Where possible connected to wired infrastructure; otherwise use of second radio interface for mesh type networking/wide area access
ORBIT: Hardware

**Hardware Components:**

- **CPU:** VIA C3 1Ghz
- **RAM:** 512 MB
- **Disk:** 20 GB
- **Power Supply:** 110 VAC
- **Ethernet:**
  - Gigabit Ethernet (control)
  - Gigabit Ethernet (data)
- **Network:**
  - Bluetooth USB
  - Intel/Atheros miniPCI 802.11 a/b/g
  - Rabbit Semi RCM3700 10 BaseT Ethernet (CM)
- **Miscellaneous:**
  - pwr/reset
  - volt/temp
  - RJ11 NodelIdBox
  - +5v standby
Experimental Evaluation

- Data provided by Adolfo/Hector

- Aggregation services:
  - Complex queries issued from each of the three deployment environments
    - Post(34-444, temp*, retrieve(count))

- Effectiveness of in-network aggregation
  - Aggregate range queries
  - Number of messages processed in the network

- Robustness in case of single peer failures

- Note that the performance of Squid and AR have been separately evaluated
Scalability of the Aggregation Services

Aggregate query: \texttt{post(<(100-300, 75-125), temperature>, retrieve(avg, 300))}

- Experiences/issues with ORBIT deployment
  - Limitations of the MAC layer/protocol (802.11x)
    - Channel interference, dropped messages, dropped beacons
  - Overlay structure is critical
    - Hierarchical, multiplexed channels
  - Message sizes and volumes impact latency
    - Average latency on 200 nodes (dedicated): \(\sim2\text{-}3\) seconds
Effectiveness of In-Network Aggregation

- Simulation results
  - 500 nodes

- AR aggregate message:
  - Post(<100-230, 35-120>, retrieve(\text{avg}))

- Fig (a) number of messages with/without in-network aggregation

- Fig (b) actual number of messages per peer
Robustness of Aggregation Service

- Assume single node failure
  - An intermediate node fails at about 205 time ticks
  - On-going aggregation operations are lost at this node
- About 100 ms required to correct the aggregation trie in a LAN environment
Conclusion

• Pervasive Computational Ecosystems & Next Generation Scientific Investigation
  – Knowledge-based, data and information driven, context-aware, computationally intensive
  – Project AutoMate: Enabling Autonomic computational science

• Meteor middleware for sensor-based pervasive Grid environments
  – Unified programming abstractions for content-based decoupled interactions and decentralized in-network aggregation
    • Recurrent and spatially constrained queries
    • Scalable and efficient trie-based design and implementation
  – Deployments on LAN, Orbit and PlanetLab
    • Scalability, performance, robustness

• More Information, publications, software
  – www.caip.rutgers.edu/~parashar/
  – parashar@caip.rutgers.edu