Rover: A Toolkit for Mobile Information Access

Anthony D. Joseph, Alan F. deLespinasse, Joshua A. Tauber, David K. Gifford and M. Frans Kaashoek

MIT Laboratory for Computer Science

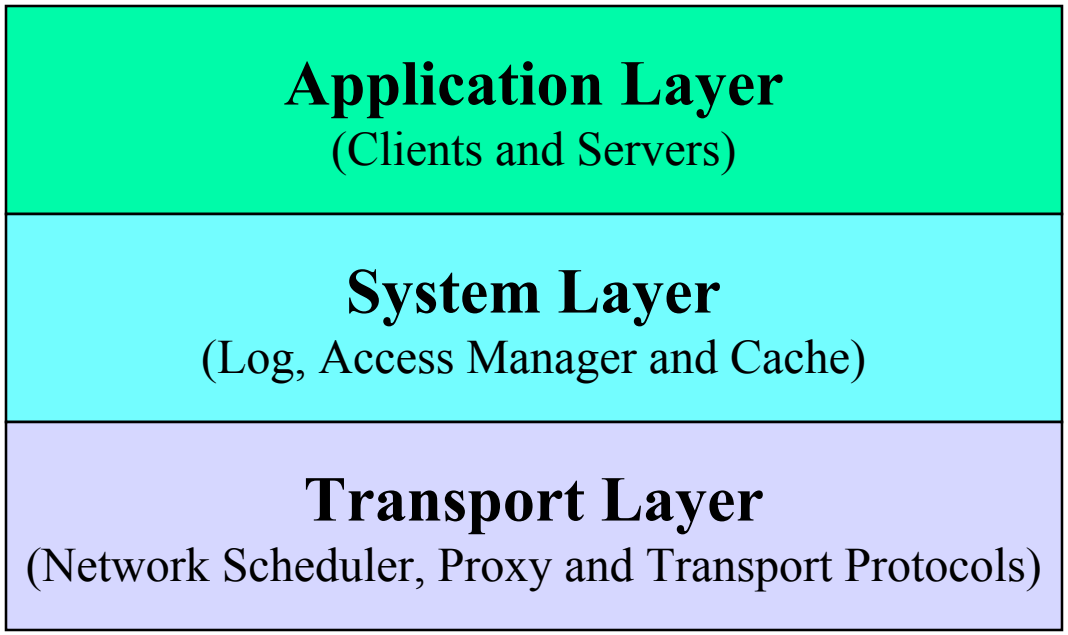
Outline

- Overview
- Architecture
- Implementation details
- Rover applications
- Performance
- Summary

Overview

- Mobile communication support for “roving” client/server applications
  - » Application-aware adaptation
  - » Application-transparent adaptation
- Two main mechanisms
  - » RDO (Relocatable Dynamic Objects)
    - – Distributed object oriented abstraction
    - – Object plus a well defined interface
  - » QRPC (Queued Remote Procedure Call)
    - – Asynchronous communication (Non-blocking RPC)
    - – Allow disconnected operation

Rover Model

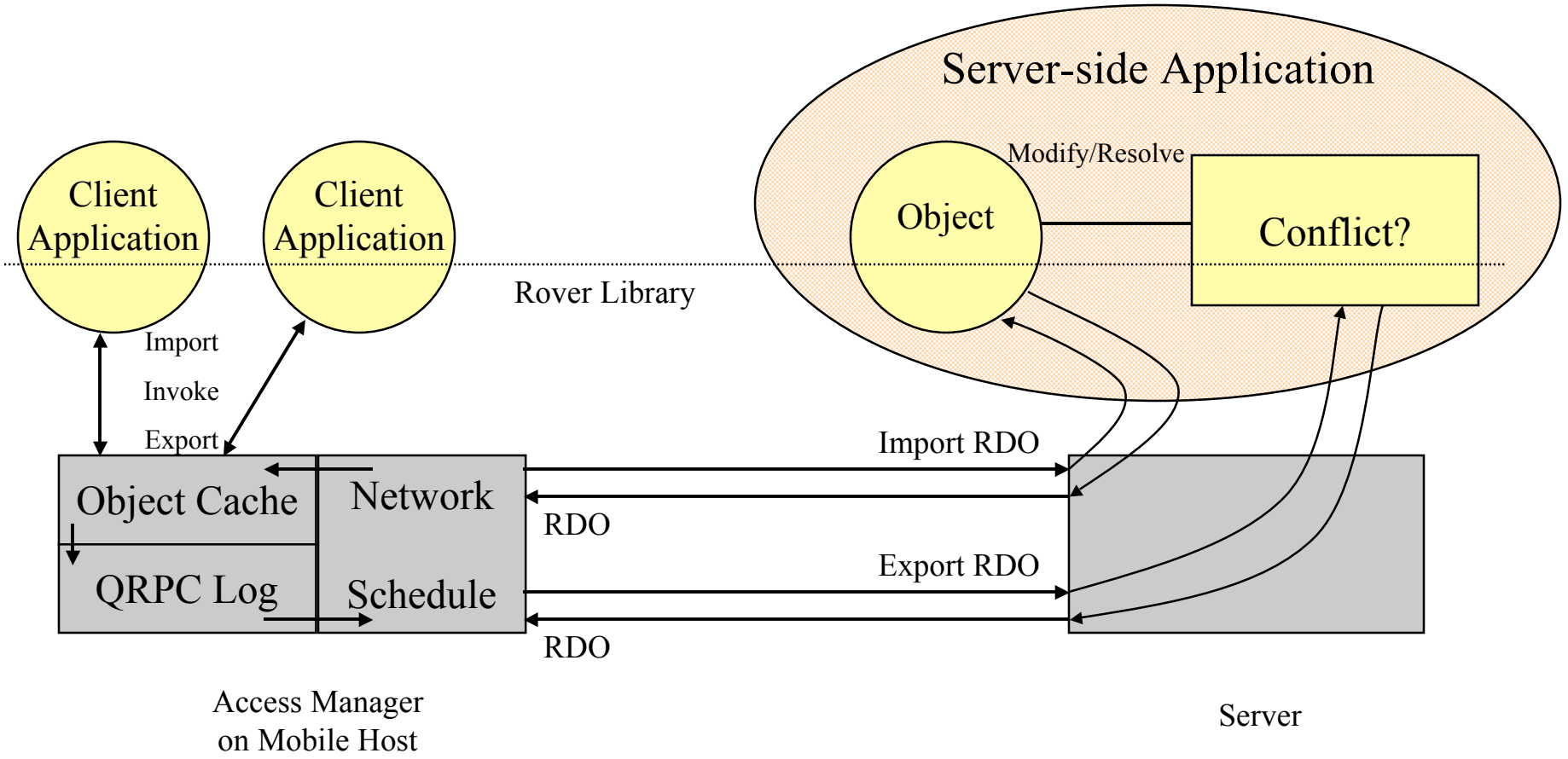


- Disconnected operation
- Load balancing
- Efficient use of available bandwidth
- Application involvement in communication decisions
- Good development environment

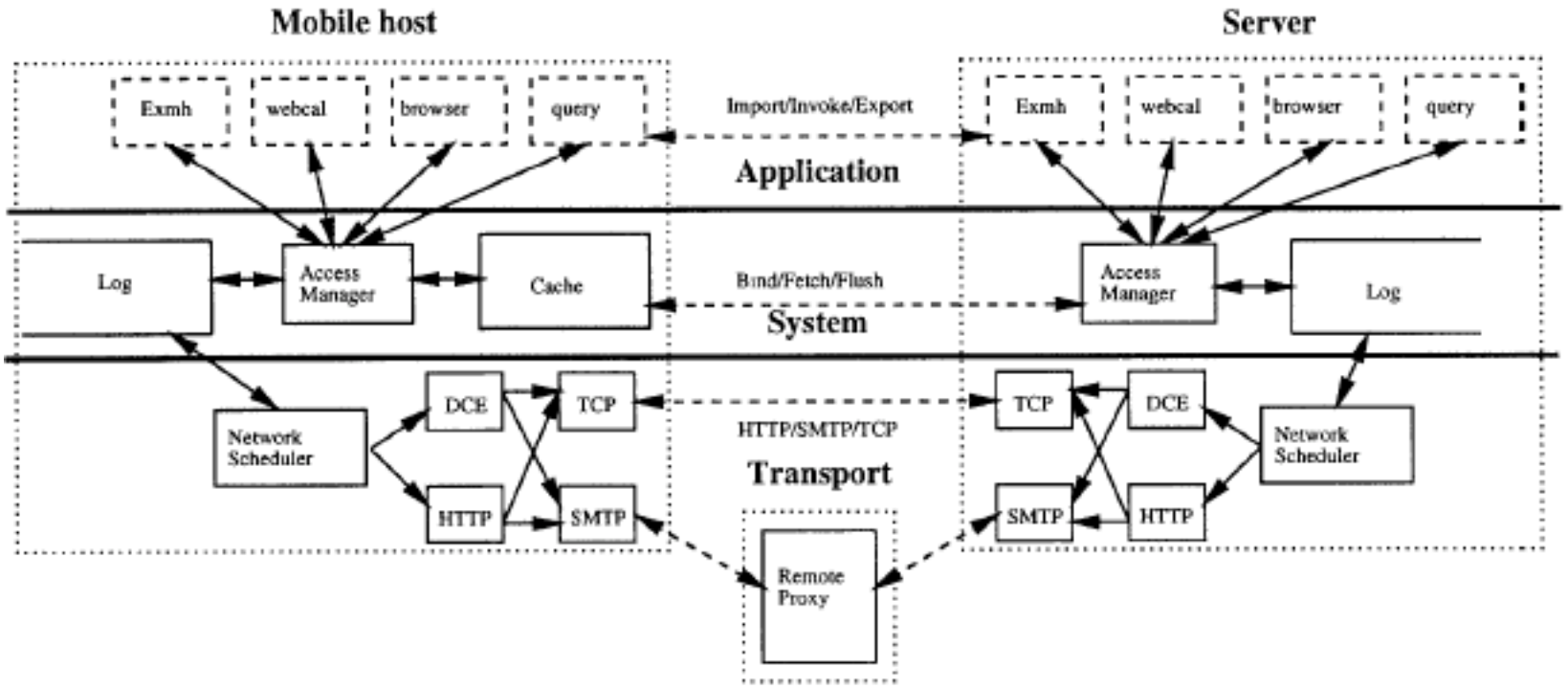
Relocatable Objects

- Mobile-aware applications can move functionality dynamically
  - » Reduces bandwidth consumption
  - » Interface at client side
  - » Heavy computation at server's side
- Caching
  - » Primary copies canonical are resident at server side (home)
  - » Secondary copies are cached at client side
  - » Objects are exported back to maintain consistency
- Support asynchronous communication
  - » Clients use QRPC to lazily fetch RDOs from servers
  - » QRPC are stored in client stable log
  - » Objects are marked tentatively committed after modification
  - » Rover scheduler drains the log in the background
  - » Lazily propagated to Rover servers

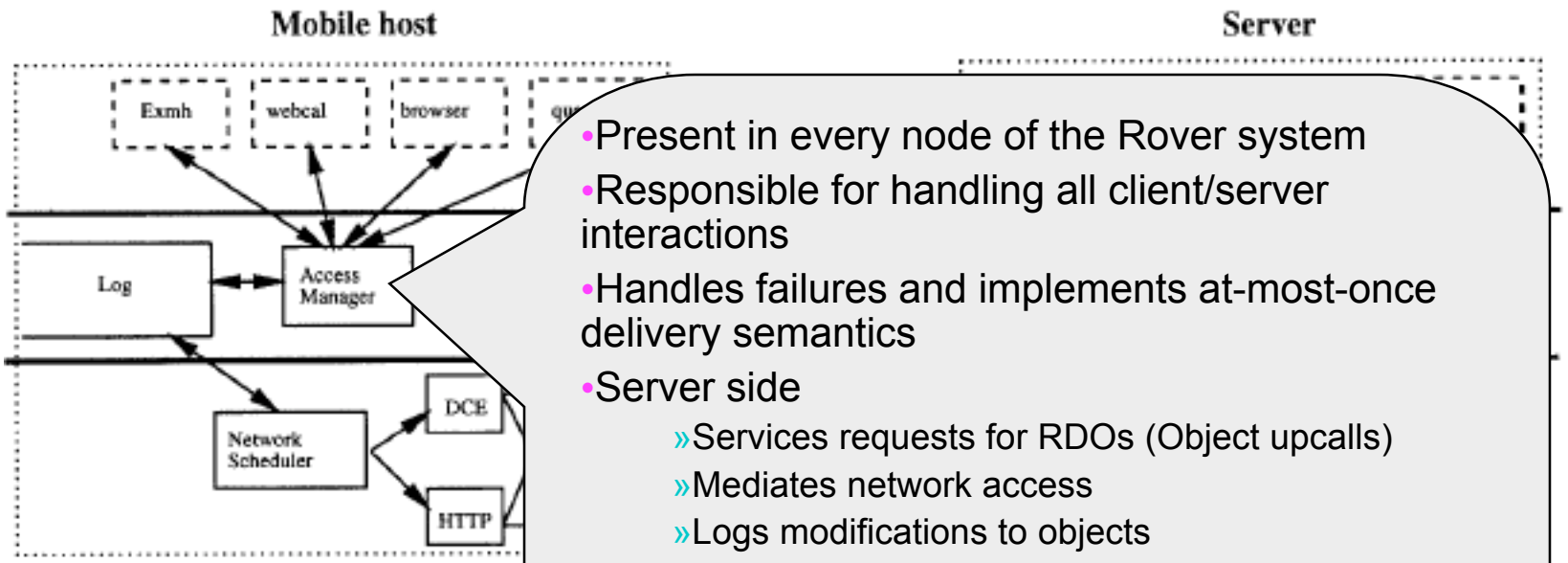
RDO Architecture



Rover Component Architecture

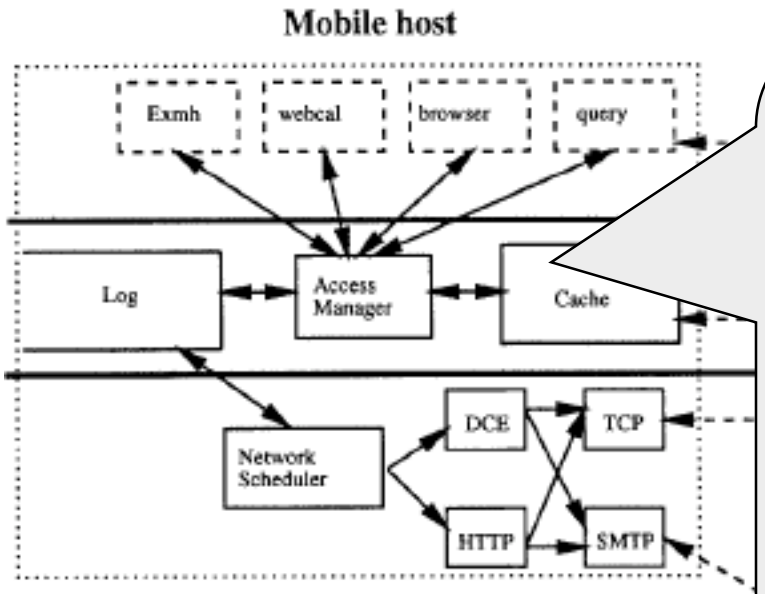


Rover Component Architecture



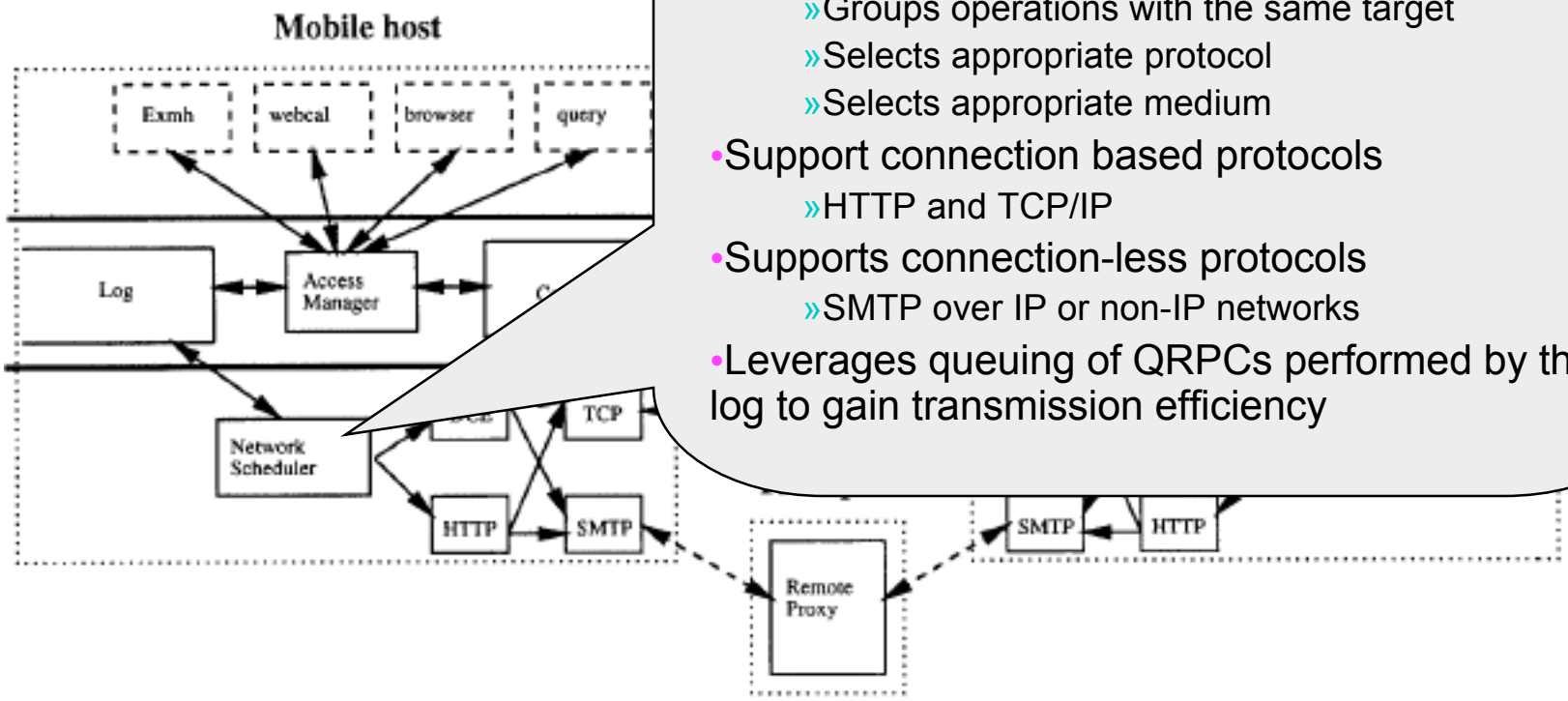
- Present in every node of the Rover system
- Responsible for handling all client/server interactions
- Handles failures and implements at-most-once delivery semantics
- Server side
  - Services requests for RDOs (Object upcalls)
  - Mediates network access
  - Logs modifications to objects
- Client side
  - Manages object cache
    - RDOs imported to object cache
    - QRPC exported to operation log
  - Mediates object invocations to server and among clients

Rover Component Architecture



- Provides stable storage for locally stored RDOs
- Two components
  - »Application private cache (App address space)
  - »A globally shared cache (Within access manager)
- Clients access log through the access manager
- If and object is found in the cache
  - »Consistency object option verified by Rover
  - »Method invocations without side-effects are processed locally
  - »Tentative data is inserted in the object cache

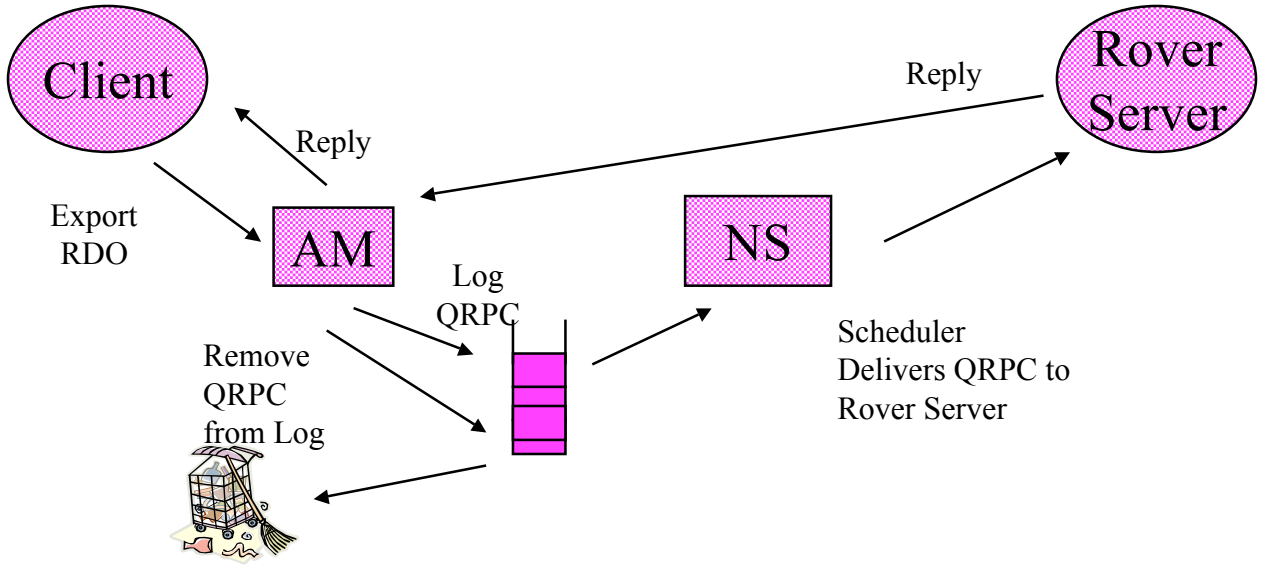
Rover Component Architecture



- Drains the operation log back to Rover servers
- Contributes to log transmission optimization
  - »Groups operations with the same target
  - »Selects appropriate protocol
  - »Selects appropriate medium
- Support connection based protocols
  - »HTTP and TCP/IP
- Supports connection-less protocols
  - »SMTP over IP or non-IP networks
- Leverages queuing of QRPCs performed by the log to gain transmission efficiency

Operation Log

- Stores QRPC
- Operation with side-effects generate QRPCs
- Incrementally flushed back to the server



Building a Rover Application

- Split the application into components
  - » Identify which components are where
- Appropriately encapsulate application's state within the objects
  - » Add an object layer
- Add support for interacting with the environment
- Add support for prefetching
  - » Decide on mechanisms to notify users of status of displayed data
- Add application-specific conflict resolution procedures

Implementation details

- Simplicity over performance
- Testbed
  - Laptop as clients (IBM ThinkPad 350C and 701C)
  - Workstations as servers (DECstation and SPARCstation)
  - 10 Mbps Ethernet, 2 Mbps WaveLAN and dial-up lines
- Application layer
  - Tcl/Tk scripts and binary applications
  - Library provides link to the access manager
  - Servers construct Tcl/Tk RDOs
  - Clients construct modification operations
  - Limited security

Rover applications

- Exmh
  - Brent Welch's Tck/Tk E-mail browser
  - Tcl/Tk wrap adds an object-based layer
- Webcal
  - Port of Ical
  - Tck/Tk again to add an object-based layer
  - Per item based consistency
- Web HTTP proxy
  - Allows "click-ahead" of the arrived data
  - SMTP-based transport layer
  - NTTP proxy
- Irolo
- Stock Market Watcher

Performance

Protocol	Transport	Latency	Throughput
TCP	Ethernet	47	0.74
	WaveLAN	61	0.48
	CSLIP14.4	500	0.02
	CSLIP2.4	3600	0.001
SMTP	Ethernet	5600	0.02
	WaveLAN	5800	0.02
	CSLIP14.4	11000	0.007
	CSLIP2.4	43000	0.001

Transport	Latency	Throughput
Ethernet	8	3.6
WaveLAN	13	0.92
CSLIP14.4	420	0.022
CSLIP2.4	3100	0.001

- Log performance is the bottleneck
- SMTP performs badly but is good for background transmission

Type	Null RDO	16KB RDO
Tcl/Tk	0.06	0.06
Client	3.2	3.2
LRPC	7.4	20

Protocol	Transport	Latency	Throughput
TCP	Ethernet	59	0.36
	WaveLAN	75	0.28
	CSLIP14.4	555	0.02
	CSLIP2.4	4100	0.001
SMTP	Ethernet	5600	0.02
	WaveLAN	5800	0.02
	CSLIP14.4	11000	0.007
	CSLIP2.4	44000	0.001

- Penalty for poor LRPC implementation
- Caching reduces latency and saves b/w
- This is basically comparing local to remote computation

Performance … (cont)

Environment	Transport	Time
X11R6	Ethernet	0:55
X11R6 (NFS)	CSLIP14.4	2:36
Xremote (Windows 3.1)	14.4 Kbit/s	9:08
Rover/App cached	CSLIP14.4	1:34
Rover/App cached	CSLIP2.4	7:56
Rover/Full cache	CSLIP14.4	1:06
Rover/Full cache	CSLIP2.4	1:37
Rover/Full cache	none	1:02

Environment	Transport	Time
X11R6	Ethernet	0:15
X11R6 (NFS)	CSLIP14.4	1:02
Xremote (Windows 3.1)	14.4 Kbit/s	3:20
Rover/App cached	CSLIP14.4	1:09
Rover/App cached	CSLIP2.4	10:16
Rover/Full cache	CSLIP14.4	0:33
Rover/Full cache	CSLIP2.4	1:11
Rover/Full cache	none	0:29

- Advantages of dynamic relocation
  - Local execution is faster than access over slow lines
  - When fully cached (code + data) the client can fully operate in disconnected mode
  - When not fully cached (code only), Rover is still competitive over slow lines

Implementation … (cont)

- System layer
  - Secure setuid application
  - Authenticates requests from clients
  - Mediates access to RDOs
  - Provides Tcl/Tk execution environment for clients
  - LPRC for intra client communication
  - Client cache and log implemented with UNIX files
- Transport layer
  - URN to HTTP POST message translator
    - URL + RDO request + authenticator
  - Network scheduler and communication protocols
    - Scheduler has several queues for different priorities
    - Network interface is chosen based on availability and quality\

Conclusions

- *RDO and QRPC are the building blocks for rover applications*
  - *Allow relocation of computation*
  - *Potential latency and bandwidth benefits over slow lines*
  - *Rover object-oriented abstractions are easily applicable to many static client/server applications*
- *Drawback*
  - *Client/Server division of responsibilities must be done careful in order to allow real disconnected operation advantages*
  - *Significant adaptation is necessary*

Discussion

- *Is Rover an object-oriented Coda?*
- *How applicable is it today given the Web dominated Internet environment?*
- *Dynamic relocation of objects adds a lot of adaptability to an application*
  - » *However, it needs to carry enough state to be useful*
  - » *Otherwise they become proxies*