

MSI - Manifold Server Interface

Converting Single, Concurrent
Server Software for a Cluster
Server Architecture

How It Works

Alexander Vrenios, Ph.D.

alex@DSRLab.com

MSI - Manifold Server Interface

OVERVIEW:

- Introduction:
 - Our system's major components
 - Goals for our final system
- Software
 - Design overview of a concurrent server
 - Design detail of a concurrent server
 - Software Conversion requirements
 - Architecture: initialization/operation
- Performance
 - Final external performance issues
- Conclusions
- Publication

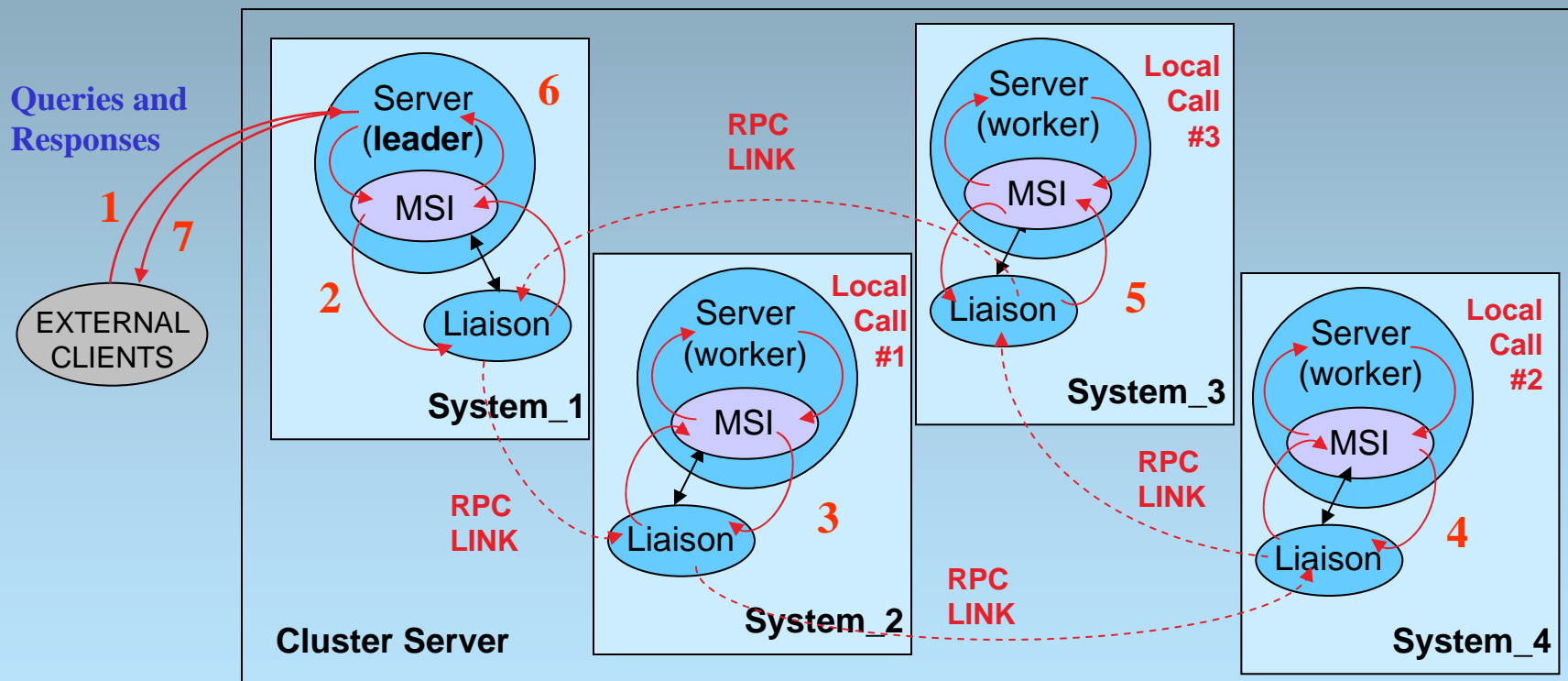
MSI - Manifold Server Interface

INTRODUCTION:

- Our System:
 - An application program interface (API) toward a reduced cost and complexity approach to high availability (HA) on a multicomputer concurrent server architecture test bed.
 - Three Components:
 - › Application provides an operational structure
 - › Input Load Generator sends transaction stream to drive the system for performance analysis
 - › Manifold Server Interface (MSI) allows quick conversion from a single system, concurrent server to that of a multicomputer cluster.
- Our Goals:
 - Reduced complexity through simplified software conversion
 - Lightweight replacement for an ORB-based infrastructure
 - Achieve HA service through software fault tolerance

MSI - Manifold Server Interface

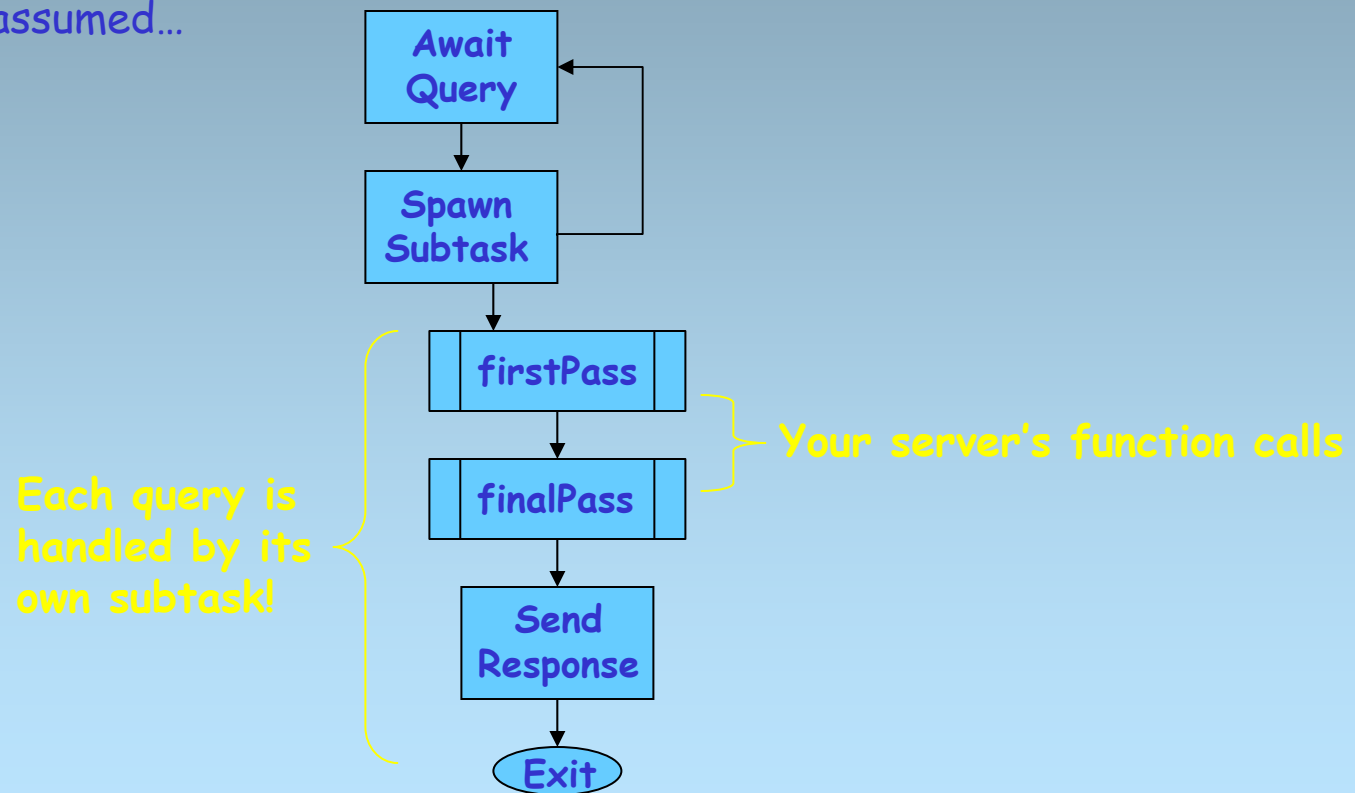
MSI SOFTWARE ARCHITECTURE:



MSI - Manifold Server Interface

SOFTWARE DESIGN OVERVIEW:

A simple concurrent server is assumed...



MSI - Manifold Server Interface

SOFTWARE DESIGN DETAIL:

- Begin with a simple concurrent server:

```
main() { /* SER.C */
    void firstPass((char*), int); /* initial query processing */
    void finalPass((char*), int); /* final query processing */

    while(1) {
        soc = accept(acc, ... );
        if((pid = fork()) == 0) { /* begin subtask */
            recv(soc, &packet, sizeof(packet), 0);
            firstPass(&packet, &code);
            finalPass(&packet, &code);
            send(soc, &packet, sizeof(packet), 0);
            exit(0); /* end subtask */
        }
    }
    void firstPass() { ... }
    void finalPass() { ... }
```

Each query is handled by its own subtask!

Your server's function calls

MSI - Manifold Server Interface

SOFTWARE CONVERSION (1 of 2):

- Modify your server's source code:
 - Tell the MSI about the functions you must call, and any global data you must support in processing:

```
#ifdef MSI
    MSI_init(firstPass, "firstPass", "0,*80,4");
    MSI_init(finalPass, "finalPass", "0,*80,4");
    MSI_init(global.number, "global.value2", 4);
    MSI_init(global.string, "global.string", *80);
    MSI_init(NULL); /* last MSI_init call */
#endif
```

INDIRECT:
pointer to
an 80 byte
value

Returns void

DIRECT:
4-byte
value

STRUCT:
4-byte
value

STRUCT (not a function) name

MSI - Manifold Server Interface

SOFTWARE CONVERSION (2 of 2):

- Modify your server's source code (continued):
 - Call your functions indirectly, through the MSI:

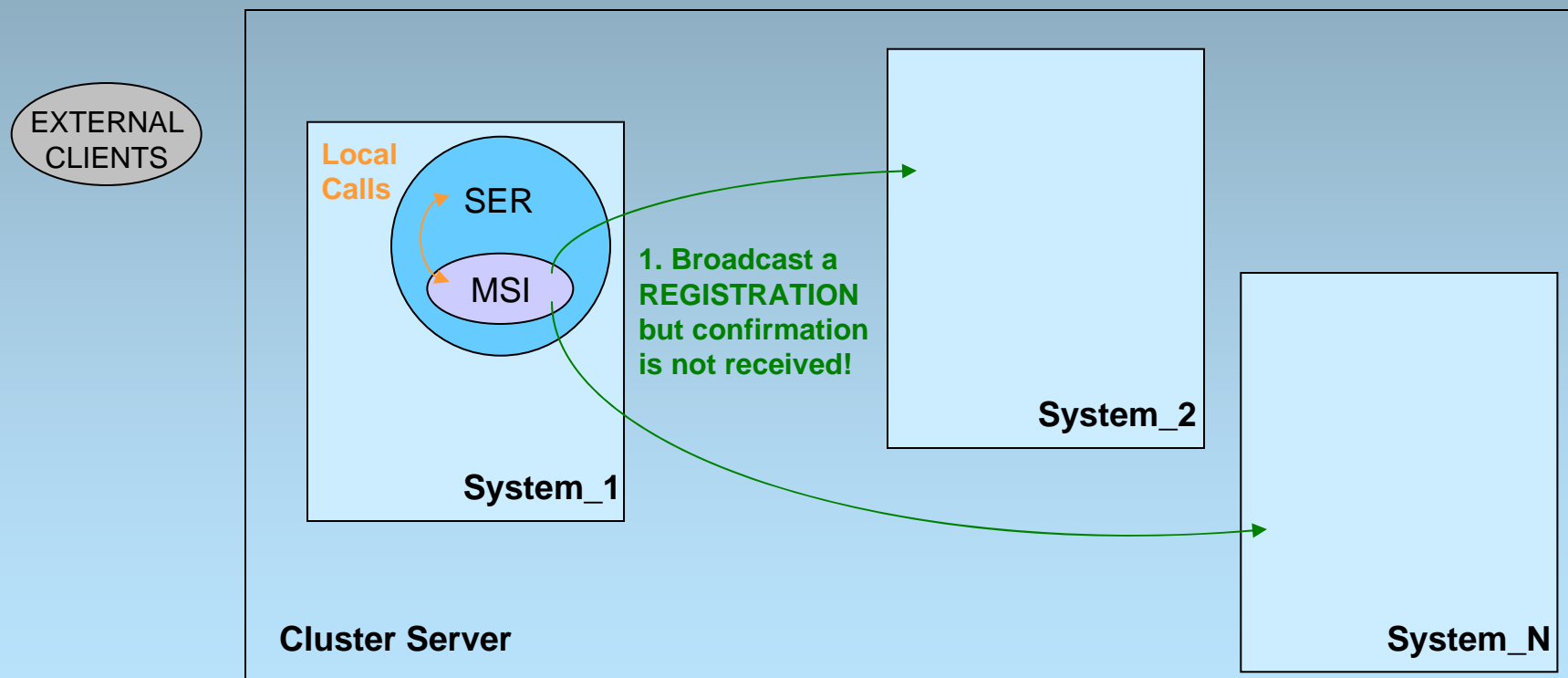
```
#ifdef MSI
    MSI_call("firstPass", NULL, &packet, &code);
#else
    firstPass(&packet, &code);
#endif
```

```
#ifdef MSI
    MSI_send(&packet, sizeof(struct packet));
#else
    send(&packet, sizeof(struct packet), 0);
#endif
```

MSI - Manifold Server Interface

SOFTWARE ARCHITECTURE:

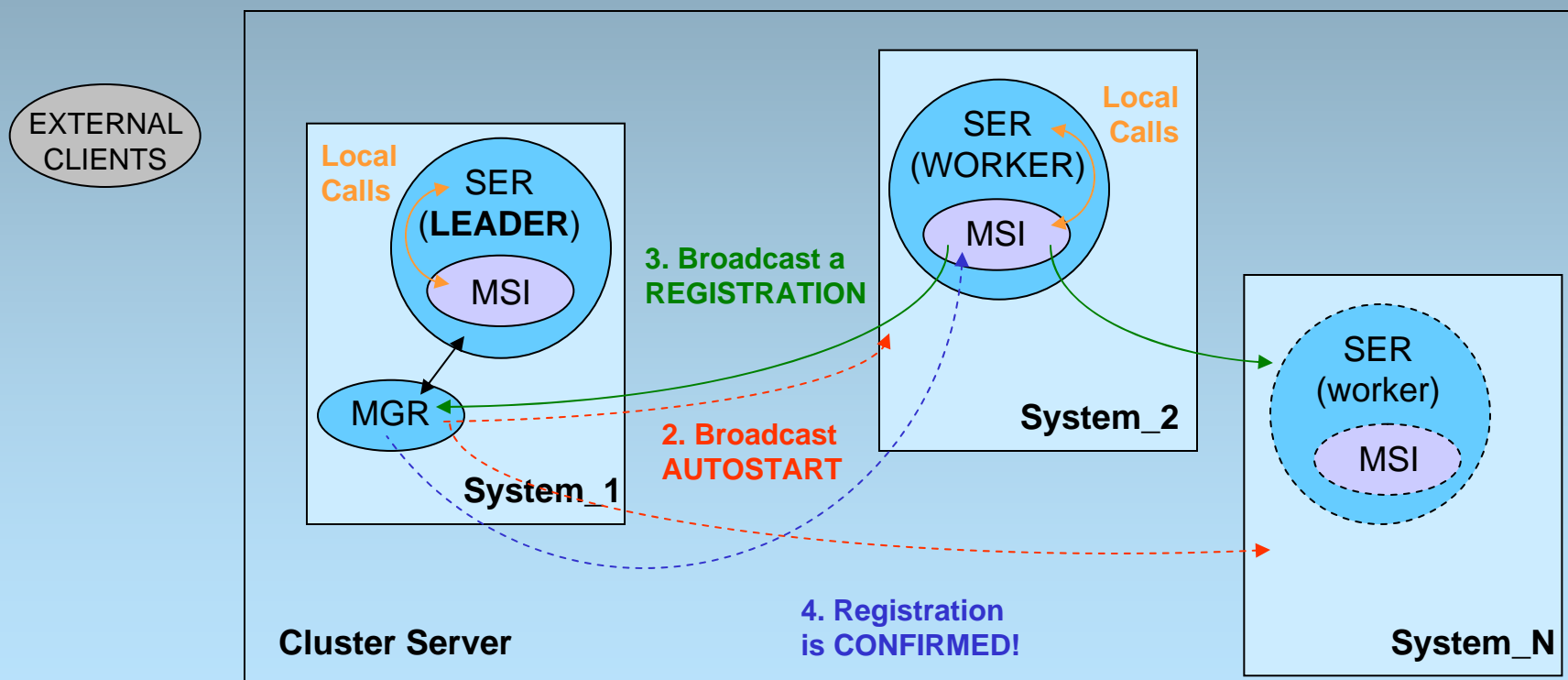
- MSI Software Initialization: Register as a new WORKER



MSI - Manifold Server Interface

SOFTWARE ARCHITECTURE:

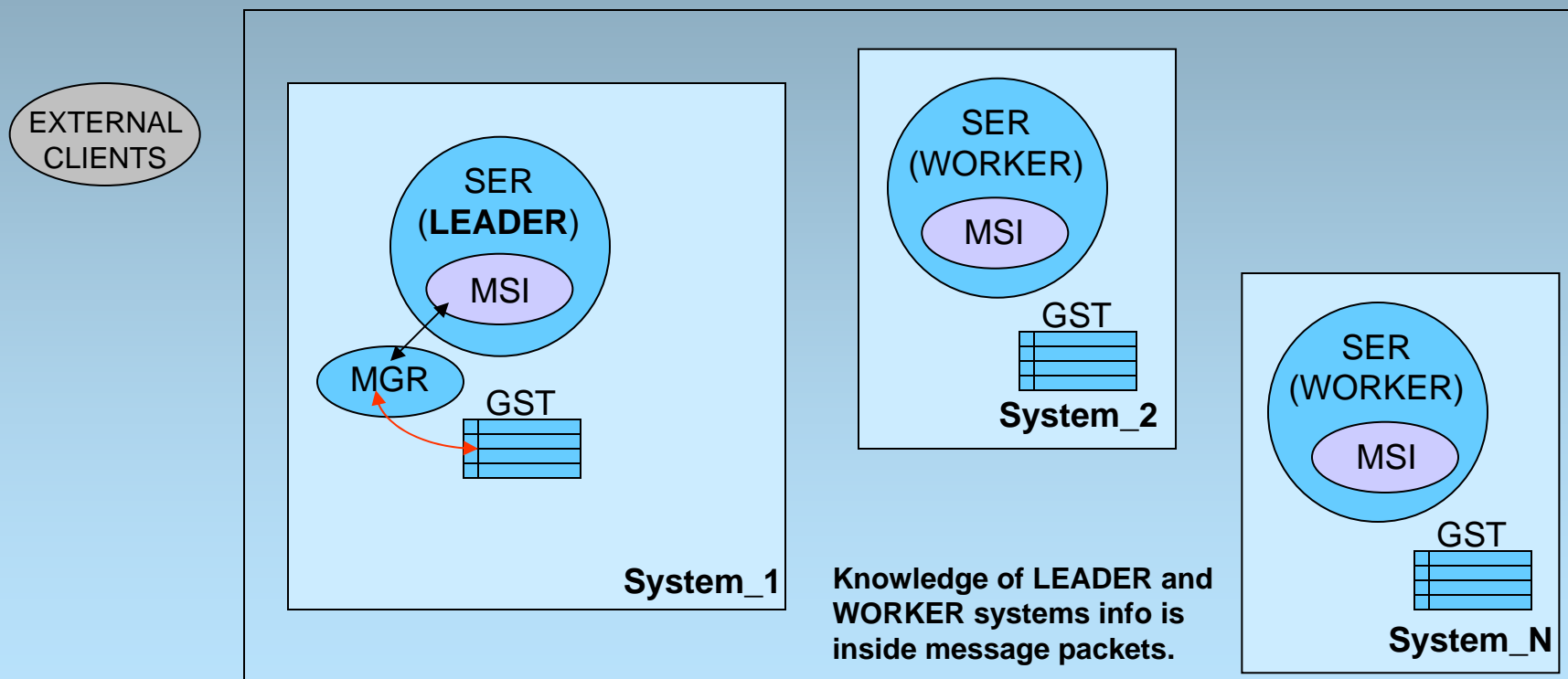
- MSI Software Initialization: Assume a LEADERSHIP role



MSI - Manifold Server Interface

SOFTWARE ARCHITECTURE:

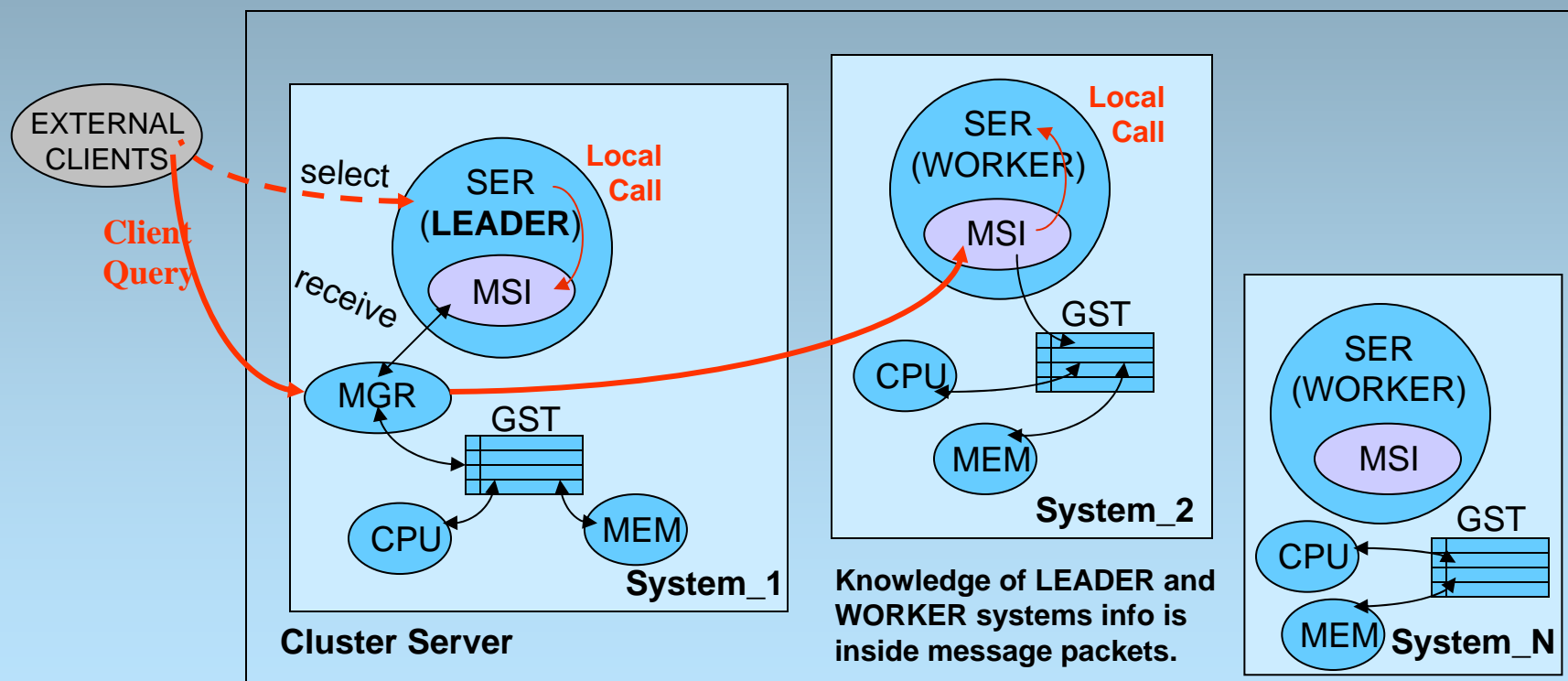
- MSI Software Initialization: Ready for client activity



MSI - Manifold Server Interface

SOFTWARE ARCHITECTURE:

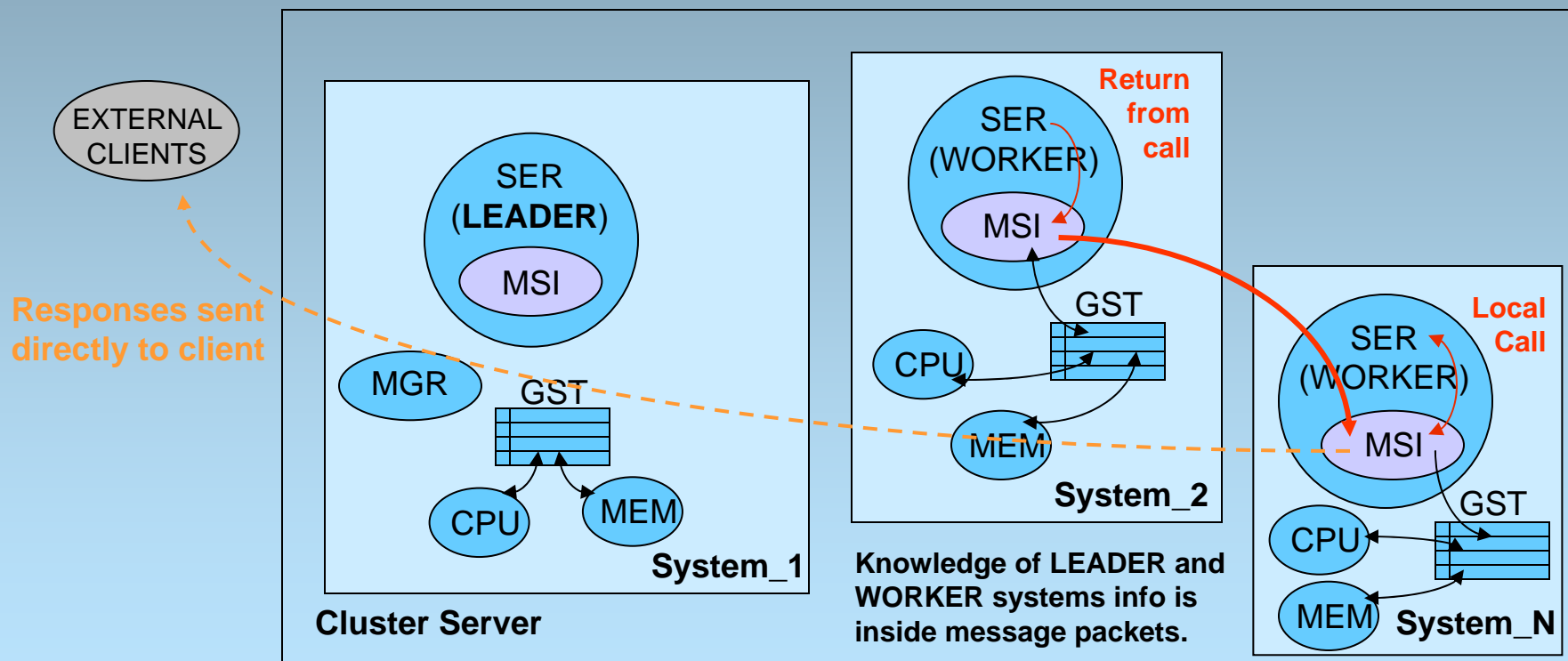
- MSI Software Operation: Remote query processing begins



MSI - Manifold Server Interface

SOFTWARE ARCHITECTURE:

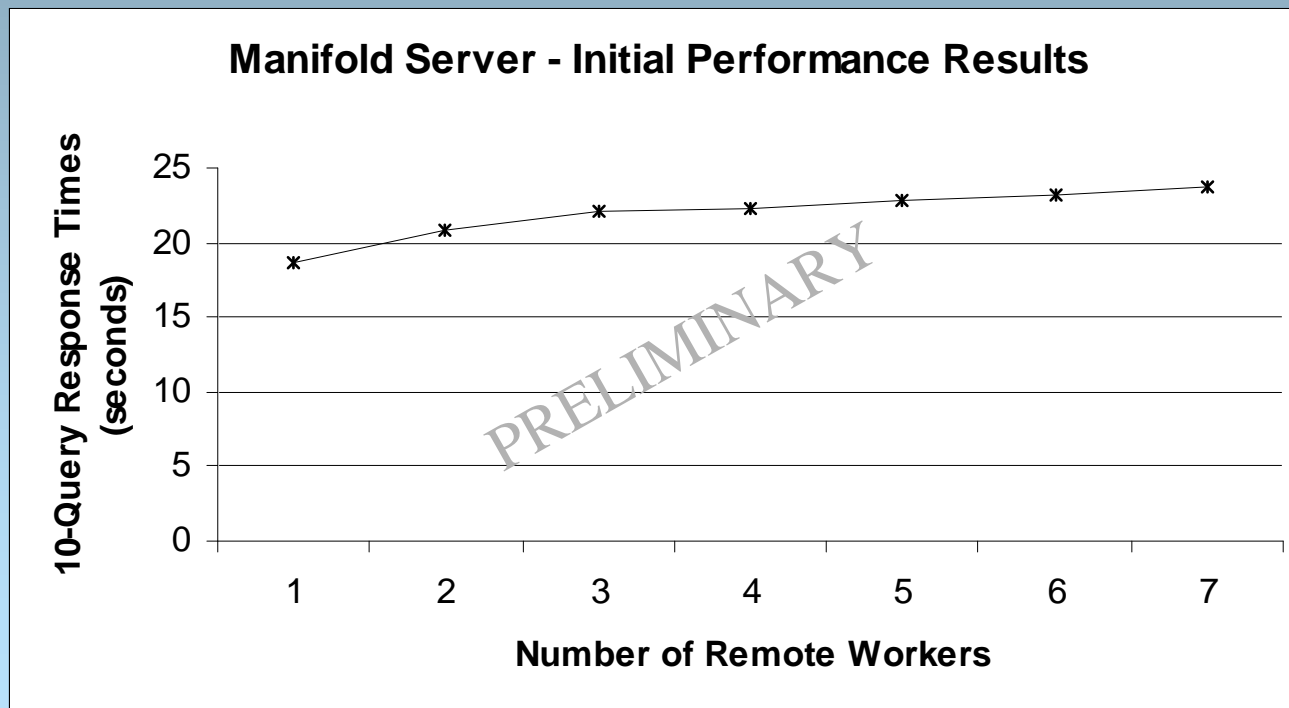
- MSI Software Operation: Remote query processing continues



MSI - Manifold Server Interface

SYSTEM PERFORMANCE:

- MSI Query Response Times - Total Times for 10 Queries:



MSI - Manifold Server Interface

SYSTEM PERFORMANCE:

- Data passing, by any other name...

Shared memory systems (IVY, Li 1989), message passing (RPC, Birrell and Nelson 1984), and event algebras (GENA, Saif, Gordon and Greaves 2001) all suffer the same kind of malady: the external I/O data pathway is MUCH slower than the internal data bus. Overall system performance is often a function of these two data rates. Tuning focuses on optimum packet size: one that strikes a balance between the number of data transfers across the external bus and the packet size.

- Our Manifold Server Interface

The MSI tacks too much data onto each message, slowing data transfer rates, limiting the MSI processing rate's upper bound.

Analyses must consider the speed and cost of such systems!

MSI - Manifold Server Interface

CONCLUSIONS:

- Low-cost upgrade from a single to a distributed server conversion
The source level software modifications are nearly trivial, and follow the design of the original application function call format.
- Straightforward system software implementation and integration
The application and the sophistication are separated by design, requiring only minor source code modifications.
- Possible lightweight replacement for ORB-based infrastructure
Maybe not yet... The system overhead is still pretty high, but there is certainly some potential in these ideas.

PUBLICATION:

"Converting Software from a Single System, Concurrent Server to a Cluster Server Architecture," *Proceedings 5th International Conference On Principles Of Distributed Systems - OPODIS 2001*, Manzanillo Mexico, Dec 9-12, 2001.