CRAY The Supercomputer Company

Architecture Up Close

Gail Alverson
Cray Inc.

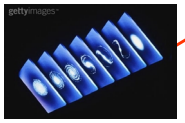
Who am I?

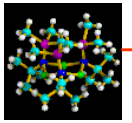
- Gail Alverson
- Lurker in your Winter 403 class
- Computer Science Professor
- Senior Engineering Manager at Cray Inc.
- Your entertainer for the next 50 minutes!

Outline

- Part 1 – Case Study: Story of an engineer
  - What's my software engineering history?
- Part 2 – Architecture up close at Cray
  - Some examples
  - Some challenges
  - Some lessons learned

Cray Inc. Focus on systems to solve huge computationally intense problems





Government-Classified
Government-Research
Weather/Environmental
Automotive
Life Sciences
Aerospace
Academic Research

Cray Solutions for:
Security
Design Engineering
Environmental
Life Sciences
Data Management/SAN

My SW Engineering roles

Software Engineer – libraries and debugger

Project lead/Software engineer, libraries and tools

Manager, Programming Environments (PE)

Senior Manager, OS and PE components

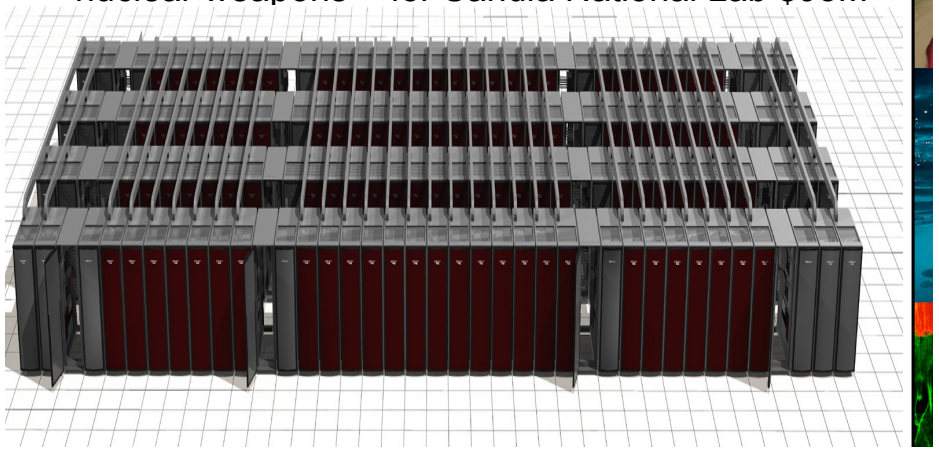
Technical Manager for the VP of Engineering

TIME

Can you think of a different SW career track?

My last project: Red Storm system

- Massively parallel processing supercomputer system used for analysis and stewardship of nuclear weapons - for Sandia National Lab $93M



System requirements included

- 10,000 AMD processors
- High speed custom interconnect
- LOW communication latency
- Fast custom microkernel on compute nodes
- Unix OS on service nodes
- High performance IO
- 2 ½ year development timeline
- No single point of failure

Can you think of some SW challenges? Risks?

How did we architect this beast?

Identify major components

While (1) {

Understand your component (and overall) requirements

Design major interfaces with dependent components

== interact with other architects

Architect your component *(the what)*

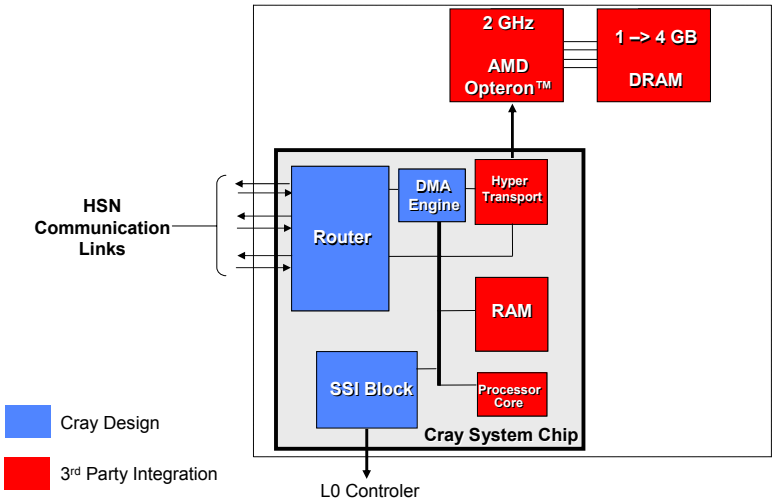
Design review internally and with customer

[Prototype/further design *(the how)*]

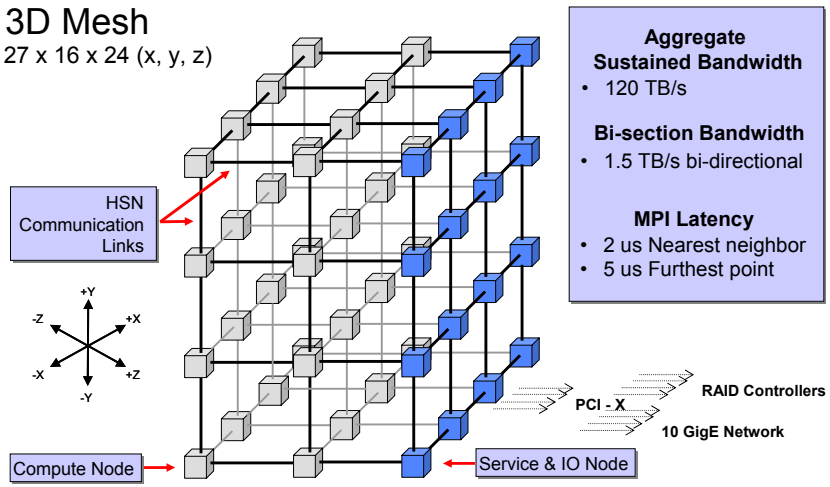
}

Here are some examples of what an architecture can capture and ways it can be presented

Hardware Architecture

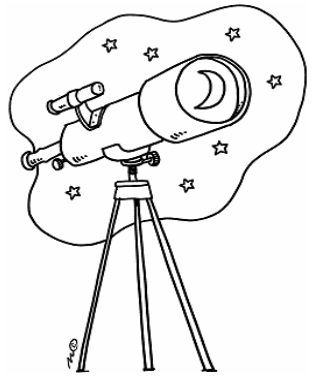


Network Architecture



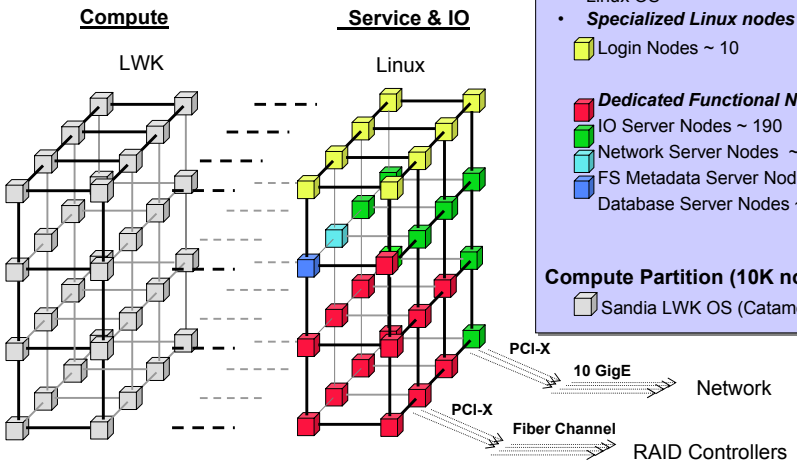
Can you remember some ways to view SW architecture?

- Code oriented views
  - Module view
  - Software layer view
- Execution oriented views
  - Data-flow view
  - Operational view
- Deployment views



Software Architecture (SA)

What type of view is this?

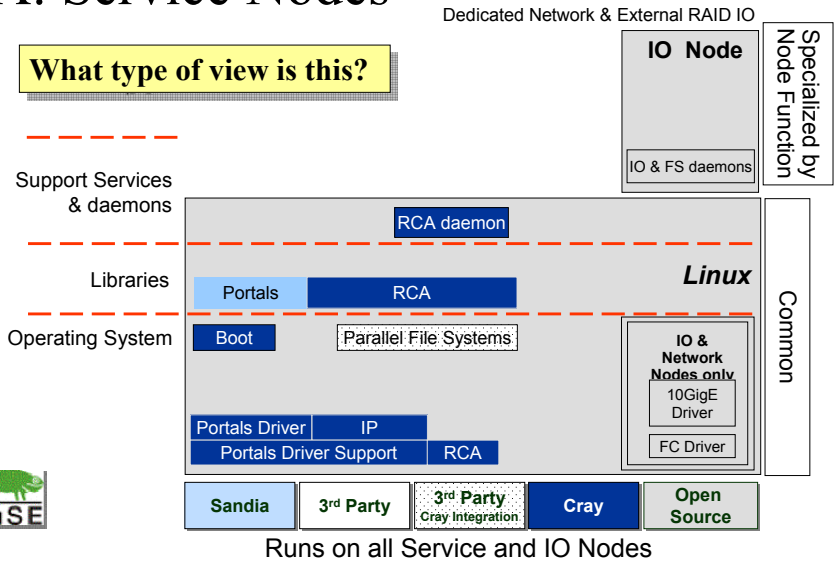


Service Partition (256 nodes)
- Linux OS
- ***Specialized Linux nodes***
  - Login Nodes ~ 10
  - ***Dedicated Functional Nodes***
  - IO Server Nodes ~ 190
  - Network Server Nodes ~ 50
  - FS Metadata Server Nodes ~ 4
  - Database Server Nodes ~ 2

Compute Partition (10K nodes)
- Sandia LWK OS (Catamount)

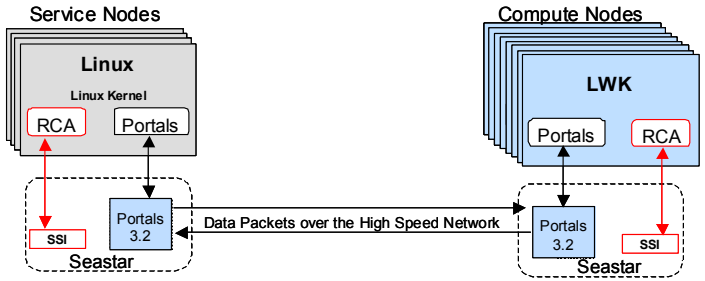
SA: Service Nodes

What type of view is this?



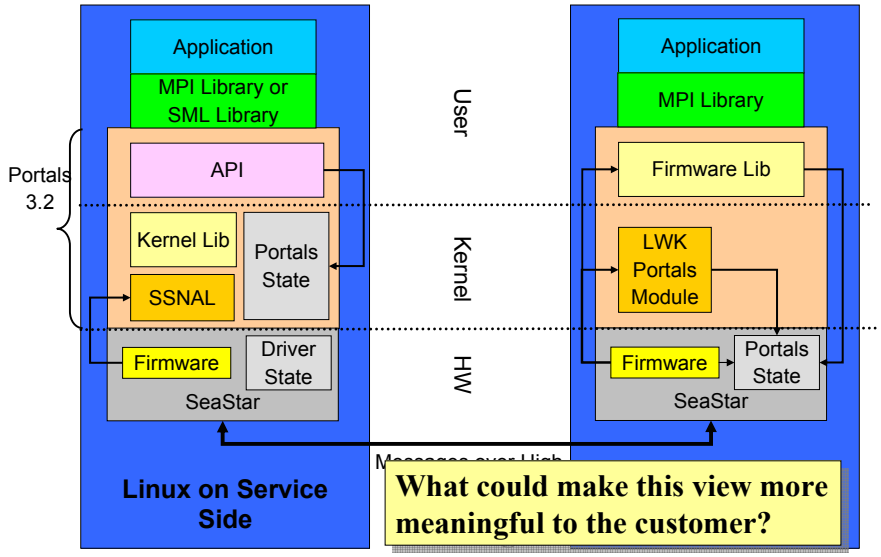
Runs on all Service and IO Nodes

SA: Message Passing



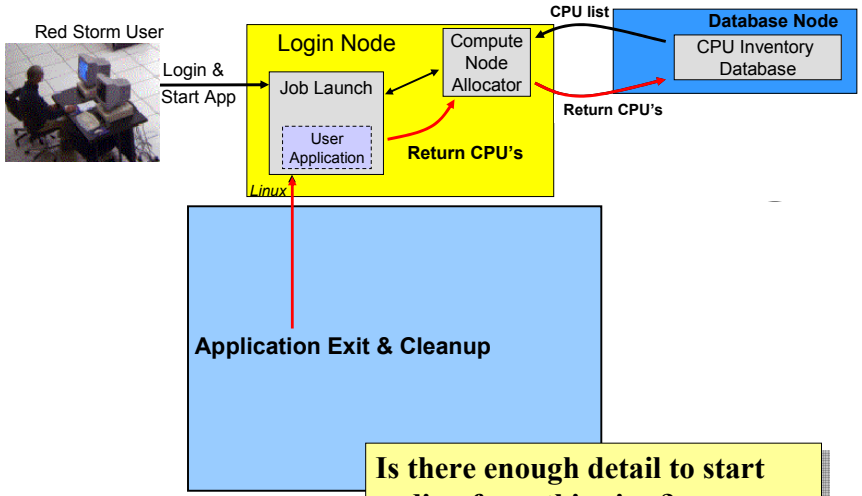
What type of view is this?

SA: more Message Passing



What could make this view more meaningful to the customer?

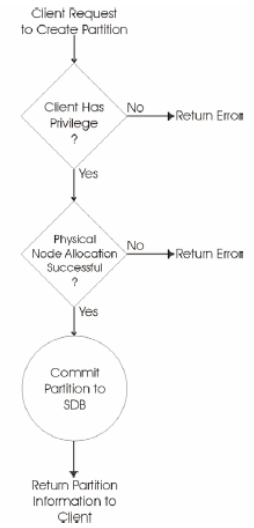
SA: Job Launch



Is there enough detail to start coding from this view?

Diagrams rock, but you need to specify the APIs as wel

Flowcharts help define the usage



cpa_create_partition

```
int cpa_create_partition(cpa_node_req_t *node_req, int job_type,
                         cpa_partition_id_t *partition_id,
                         cpa_cookie_t *admin_cookie,
                         cpa_cookie_t *alloc_cookie)
```

Description:
This function sends a "create partition" request to the CPA daem
caller.

Arguments:

cpa_node_req_t	* node_req	IN/OUT:	list of node success, ea nodes alloc
int	job_type	IN:	the type of being creat CPA_INTERACTIVE or CPA_PBS
cpa_partition_id_t	*partition_id	OUT:	ID of the partition created
cpa_cookie_t	*admin_cookie	OUT:	partition's administration cookie
cpa_cookie_t	*alloc_cookie	OUT:	partition's allocaiton cookie

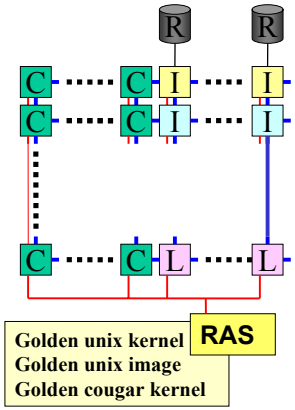
Returns:

CPA_OK	Success
CPA_INVAL_ARGS	Failure, caller passed invalid arguments
CPA_GONE	Failure, CPA daemon couldn't be contacted
CPA_NO_NODES	Failure, not enough nodes available

API's specify enough of the "what" for the next development phase, of designing the "how"

One last example of a SA: System boot

Requirement 5.11 The full system must be able to be booted in less than 15 minutes after a clean shutdown.



Service Nodes

1. RAS broadcasts kernel to Service nodes
2. Service nodes verify Unix image is valid
   If not, first loads image from RAS and others, load from peer
3. Service nodes are up and running

Compute Nodes

1. RAS broadcasts kernel to cage controllers
2. Cage controllers broadcast to other nodes
3. Compute nodes are up and running

How is software different?

Pictures tend to be part of an overall architecture document

- **Introduction**
  - **Revision history!**
  - **Scope**
  - **Requirements**
- **System description**
- **Deployment view**
  - **Intro**
  - **Top level diagram**
  - **Interfaces**
  - **Issues to resolve (risks, mitigations)**
- **Functional view**
- **Software layer view**
- **…**

Can you think of other elements?

CRAY The Supercomputer Company

Stepping back, what was important to capture in the architecture?

Can you see any common themes?

- Multiple views of the components
- Diagrams, diagrams, diagrams
- Focus on interfaces and their requirements

the "what" of the integration points

CRAY The Supercomputer Company

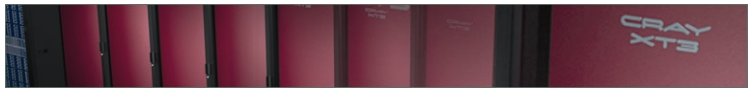
What was most important in the architecture process?

What do you think?

- Reviews
  - Both internal and with the customer
- Iteration

CRAY The Supercomputer Company

Epilogue



- Red Storm was made into a product, Cray XT3
- Full delivery was 3 ½+ years, but got something to the customer in 3
- Software effort was much more complex than expected
- Rearchitected at least two major SW components after getting experience with them
- Product is successful and in demand (yah!)