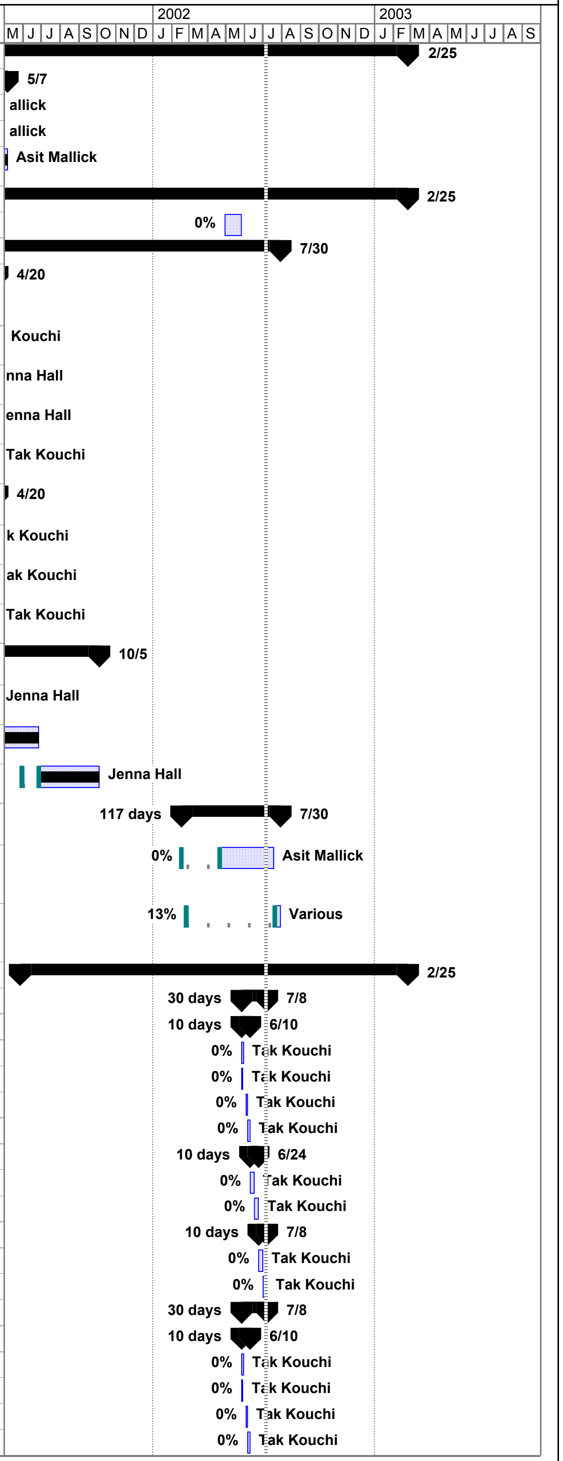
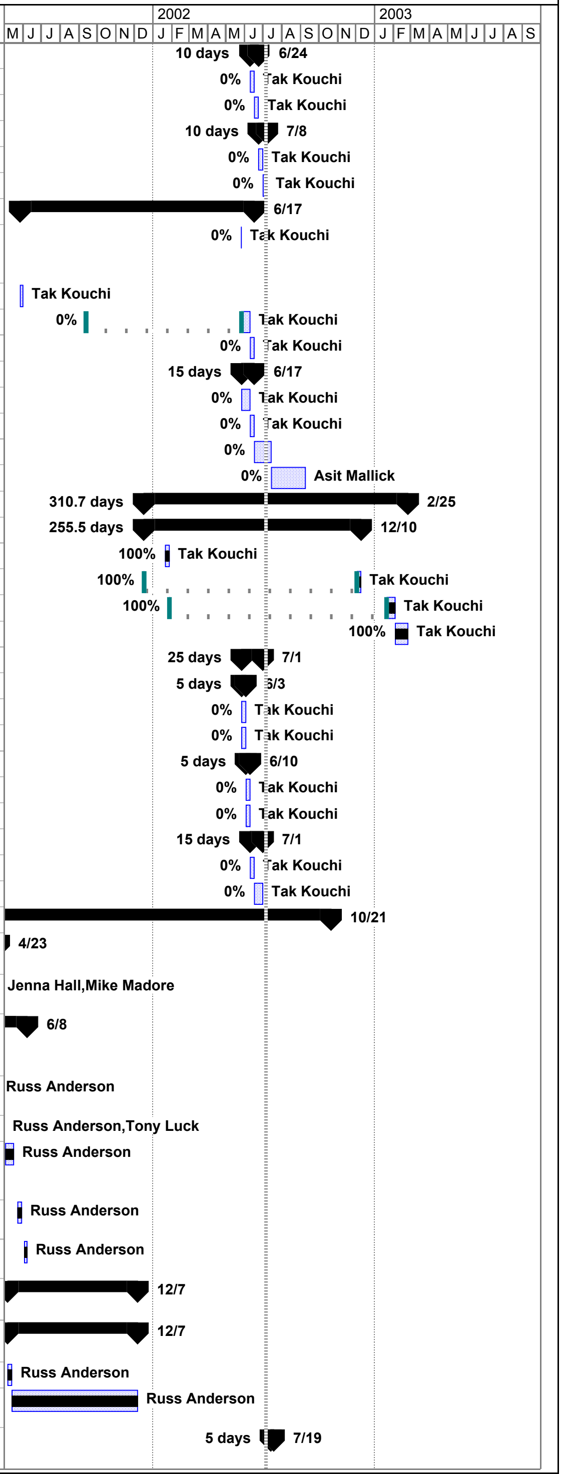
Atlas Task List

ID	ℹ	ID	Task Name	Pri	Duration	% Complet	Start	Finish	Predecessors	Resource Names
1			Overall Project		537.7 days?	51%	Fri 2/2/01	Tue 2/25/03		
2	✓	1	Basic Boot Support	1	66 days	100%	Mon 2/5/01	Mon 5/7/01		
3	✓	1.1	ACPI 2.0 table support	1	5 days	100%	Mon 2/5/01	Fri 2/9/01		Asit Mallick
4	✓	1.2	Remove Itanium implementation specific details	1	5 days	100%	Mon 2/5/01	Fri 2/9/01		Asit Mallick
5	✓	1.3	Power on debug - PAL/SAL implementation workarounds (ongoing task)	1	66 days	100%	Mon 2/5/01	Mon 5/7/01		Asit Mallick
6		2	ACPI 2.0 Support for Enabling Large Systems	1	521.7 days?	47%	Mon 2/26/01	Tue 2/25/03		Asit Mallick[55%]
7	▦		Firmware specification document provided by NEC		20 days	0%	Tue 4/30/02	Mon 5/27/02		
8			Initialization coding, testing		372 days	75%	Mon 2/26/01	Tue 7/30/02		Tak Kouchi
9	✓	2.1	Initialization sequence (to support cc-NUMA) - Interface with discontig/bootmem memory support	1	40 days	100%	Mon 2/26/01	Fri 4/20/01		Tak Kouchi
10	✓	2.1.1	Design of interface for memory initialization	1	10 days	100%	Mon 2/26/01	Fri 3/9/01		Tak Kouchi
11	✓	2.1.2	SRAT table parsing	1	20 days	100%	Mon 2/26/01	Fri 3/23/01		Jenna Hall
12	✓ 🗒	2.1.3	Simulation of SRAT tables for testing	1	10 days	100%	Mon 3/26/01	Fri 4/6/01		Jenna Hall
13	✓	2.1.4	Testing of SRAT tables on physical hardware	1	10 days	100%	Mon 4/9/01	Fri 4/20/01	12	Tak Kouchi
14	✓	2.2	SLIT as an alternative to _PXM	1	30 days	100%	Mon 3/12/01	Fri 4/20/01		Tak Kouchi
15	✓	2.2.1	SLIT table parsing	1	10 days	100%	Mon 3/12/01	Fri 3/23/01		Tak Kouchi
16	✓ 🗒	2.2.2	Simulation of SLIT tables for testing	1	10 days	100%	Mon 3/26/01	Fri 4/6/01		Tak Kouchi
17	✓	2.2.3	Testing of SLIT tables on physical hardware	1	10 days	100%	Mon 4/9/01	Fri 4/20/01	16	Tak Kouchi
18		2.3	Interrupt routing APIs for device drivers - provide capability to write device drivers special devices	1	130 days	99%	Mon 4/9/01	Fri 10/5/01		Jenna Hall
19	✓	2.3.1	API development	1	10 days	100%	Mon 4/9/01	Fri 4/20/01		Jenna Hall
20	✓	2.3.2	Development of drivers in order to do interrupt routing testing	1	48 days	100%	Mon 4/23/01	Wed 6/27/01	19	
21	✓ 🗒	2.3.3	Testing	1	74.5 days	100%	Tue 5/29/01	Fri 10/5/01	20	Jenna Hall
22	🗒	2.4	Power driver - ACPI sleep state management (kernel). Investigate work done by other parties	1	117 days	1%	Mon 2/18/02	Tue 7/30/02		Jenna Hall
23	▦	2.4.1	Driver development - external team (ACTUAL COMPLETION DATE UNKNOWN)	1	65 days	0%	Mon 2/18/02	Fri 7/19/02		Asit Mallick
24	🗒	2.4.2	Testing on each platform. (Intel - Tiger/Lion/BigSur, NEC - AsAmA/AzusA, Others - unknown)	1	8 days	13%	Mon 2/25/02	Tue 7/30/02	23	Various
25			Advanced ACPI 2.0 coding, testing		456.7 days?	15%	Mon 5/28/01	Tue 2/25/03		
26	🗒	2.5	Processor module driver	1	30 days	0%	Tue 5/28/02	Mon 7/8/02		Tak Kouchi
27		2.5.1	Design	1	10 days	0%	Tue 5/28/02	Mon 6/10/02		Tak Kouchi
28		2.5.1.1	Interface to ACPI CA driver	1	3 days	0%	Tue 5/28/02	Thu 5/30/02	7	Tak Kouchi
29		2.5.1.2	Interface to processor hotplug driver	1	2 days	0%	Tue 5/28/02	Wed 5/29/02	7	Tak Kouchi
30		2.5.1.3	Interface to module device driver	1	3 days	0%	Tue 6/4/02	Thu 6/6/02	67	Tak Kouchi
31		2.5.1.4	Planning, testing	1	2 days	0%	Fri 6/7/02	Mon 6/10/02	30	Tak Kouchi
32		2.5.2	Implementation	1	10 days	0%	Tue 6/11/02	Mon 6/24/02	27	Tak Kouchi
33		2.5.2.1	Driver implementation	1	5 days	0%	Tue 6/11/02	Mon 6/17/02	7	Tak Kouchi
34		2.5.2.2	Implementation of emulation routines	1	5 days	0%	Tue 6/18/02	Mon 6/24/02	33	Tak Kouchi
35		2.5.3	Testing	1	10 days	0%	Tue 6/25/02	Mon 7/8/02	32	Tak Kouchi
36		2.5.3.1	Testing with emulation routines	1	5 days	0%	Tue 6/25/02	Mon 7/1/02		Tak Kouchi
37		2.5.3.2	Testing with real hardware/firmware	1	5 days	0%	Tue 7/2/02	Mon 7/8/02	36,75	Tak Kouchi
38		2.6	Memory module driver	1	30 days	0%	Tue 5/28/02	Mon 7/8/02		Tak Kouchi
39		2.6.1	Design	1	10 days	0%	Tue 5/28/02	Mon 6/10/02		Tak Kouchi
40		2.6.1.1	Interface to ACPI CA driver	1	3 days	0%	Tue 5/28/02	Thu 5/30/02	7	Tak Kouchi
41		2.6.1.2	Interface to memory hotplug driver	1	2 days	0%	Tue 5/28/02	Wed 5/29/02	7	Tak Kouchi
42		2.6.1.3	Interface to module device driver	1	3 days	0%	Tue 6/4/02	Thu 6/6/02	67	Tak Kouchi
43		2.6.1.4	Planning, testing	1	2 days	0%	Fri 6/7/02	Mon 6/10/02	42	Tak Kouchi



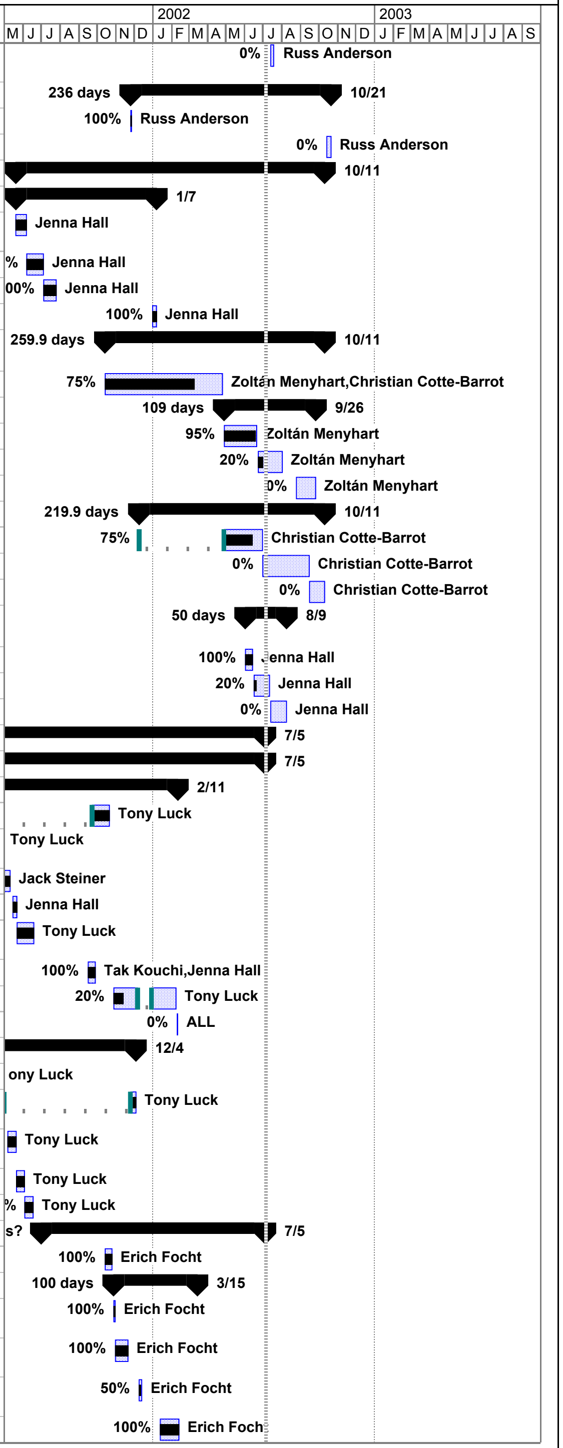
Atlas Task List

ID	ℹ	ID	Task Name	Pri	Duration	% Complet	Start	Finish	Predecessors	Resource Names
44		**2.6.2**	**Implementation**	**1**	**10 days**	**0%**	**Tue 6/11/02**	**Mon 6/24/02**	**39**	**Tak Kouchi**
45		2.6.2.1	Driver implementation	1	5 days	0%	Tue 6/11/02	Mon 6/17/02		Tak Kouchi
46		2.6.2.2	Implementation of emulation routines	1	5 days	0%	Tue 6/18/02	Mon 6/24/02	45	Tak Kouchi
47		**2.6.3**	**Testing**	**1**	**10 days**	**0%**	**Tue 6/25/02**	**Mon 7/8/02**	**44**	**Tak Kouchi**
48		2.6.3.1	Testing with emulation routines	1	5 days	0%	Tue 6/25/02	Mon 7/1/02		Tak Kouchi
49		2.6.3.2	Testing with real hardware/firmware	1	5 days	0%	Tue 7/2/02	Mon 7/8/02	48,75	Tak Kouchi
50	📝	**2.7**	**GPE block driver - Move to bottom of list.**	**3**	**276 days?**	**0%**	**Mon 5/28/01**	**Mon 6/17/02**		**Tak Kouchi**
51	📅		Firmware available which implements this functionality. ACTUAL DUE DATE UNKNOWN, 5/27/02 IS A PLACEHOLDER.		1 day?	0%	Mon 5/27/02	Mon 5/27/02		Tak Kouchi
52	📅	2.7.1	Design	3	5 days	0%	Mon 5/28/01	Fri 6/1/01		Tak Kouchi
53	📅!	2.7.2	Implementation	3	10 days	0%	Fri 9/14/01	Mon 6/10/02	52,51	Tak Kouchi
54		2.7.3	Test	3	5 days	0%	Tue 6/11/02	Mon 6/17/02	53	Tak Kouchi
55	📝	**2.8**	**Logical to physical identification for devices**	**2**	**15 days**	**0%**	**Tue 5/28/02**	**Mon 6/17/02**		**Tak Kouchi**
56	📅	2.8.1	Design/implementation	2	10 days	0%	Tue 5/28/02	Mon 6/10/02	51	Tak Kouchi
57		2.8.2	Testing	2	5 days	0%	Tue 6/11/02	Mon 6/17/02	56	Tak Kouchi
58		2.9	PCI bus segment support	3	20 days	0%	Tue 6/18/02	Mon 7/15/02	57	
59	📝	2.10	Cleanup of current ACPI support in IA64 Linux	2	40 days?	0%	Tue 7/16/02	Mon 9/9/02	58	Asit Mallick
60	✓ 📝	**2.11**	**PCI hot plug support**	**1**	**310.7 days**	**100%**	**Tue 12/18/01**	**Tue 2/25/03**		
61	✓	**2.11.1**	**Design**	**1**	**255.5 days**	**100%**	**Tue 12/18/01**	**Tue 12/10/02**		**Tak Kouchi**
62	✓	2.11.1.1	Interface to ACPI CA core driver	1	5 days	100%	Tue 1/22/02	Mon 1/28/02	7	Tak Kouchi
63	✓	2.11.1.2	Interface to PCI hotplug driver	1	5 days	100%	Tue 12/18/01	Tue 12/10/02	7	Tak Kouchi
64	✓	2.11.2	Implementation	1	10 days	100%	Tue 1/29/02	Tue 2/4/03	62	Tak Kouchi
65	✓	2.11.3	Testing	1	15 days	100%	Tue 2/4/03	Tue 2/25/03	64,51	Tak Kouchi
66	📝	**2.12**	**Module device driver**	**1**	**25 days**	**0%**	**Tue 5/28/02**	**Mon 7/1/02**		**Tak Kouchi**
67		**2.12.1**	**Design**	**1**	**5 days**	**0%**	**Tue 5/28/02**	**Mon 6/3/02**		**Tak Kouchi**
68		2.12.1.1	Interface to ACPI CA driver	1	5 days	0%	Tue 5/28/02	Mon 6/3/02	7	Tak Kouchi
69		2.12.1.2	Interface to node-hotplug driver	1	5 days	0%	Tue 5/28/02	Mon 6/3/02	7	Tak Kouchi
70		**2.12.2**	**Implementation**	**1**	**5 days**	**0%**	**Tue 6/4/02**	**Mon 6/10/02**	**67**	**Tak Kouchi**
71		2.12.2.1	Driver implementation	1	5 days	0%	Tue 6/4/02	Mon 6/10/02		Tak Kouchi
72		2.12.2.2	Implementation of emulation routines	1	5 days	0%	Tue 6/4/02	Mon 6/10/02		Tak Kouchi
73		**2.12.3**	**Testing**	**1**	**15 days**	**0%**	**Tue 6/11/02**	**Mon 7/1/02**		**Tak Kouchi**
74		2.12.3.1	Testing with emulation routines	1	5 days	0%	Tue 6/11/02	Mon 6/17/02	70	Tak Kouchi
75		2.12.3.2	Testing with real hardware/firmware	1	10 days	0%	Tue 6/18/02	Mon 7/1/02	74	Tak Kouchi
76		**3**	**Static Domain Partitioning.**		**396 days**	**95%**	**Mon 4/16/01**	**Mon 10/21/02**		**Russ Anderson**
77	📝	**3.1**	**OS support for network install boot and external install boot**	**1**	**0.5 days**	**90%**	**Mon 4/23/01**	**Mon 4/23/01**		**Jenna Hall**
78	📅		Firmware network boot in EFI (BootP). Verification that it works (Utilizing PXE)		0.5 days	90%	Mon 4/23/01	Mon 4/23/01		Jenna Hall,Mike Madore
79	✓ 📝	**3.2**	**Legacy free boot (uses one serial port) -Linux can boot with no physical memory at 0; -EFI running with no physical memory at 0**	**2**	**40 days**	**100%**	**Mon 4/16/01**	**Fri 6/8/01**		**Russ Anderson**
80	✓ 📝	3.2.1	Verify that Linux has no dependency on PC address map	2	5 days	100%	Mon 4/16/01	Fri 4/20/01		Russ Anderson
81	✓ 📝	3.2.2	Linux boot on w/o physical memory 0	2	5 days	100%	Wed 4/25/01	Tue 5/1/01	80	Russ
82	✓	3.2.3	EFI boot loader not dependent on a given fixed address. Allow kernel to be loaded at any physical address	2	10 days	100%	Fri 5/4/01	Thu 5/17/01	81	Russ Anderson
83	✓ 📝	3.2.4	Verify that there is no hidden 32 bit address dependencies in EFI boot loader	2	5 days	100%	Thu 5/24/01	Wed 5/30/01	82	Russ Anderson
84	✓	3.2.5	Platform companies verify that EFI can get loaded at any physical address (i.e. without phys addr 0	2	5 days	100%	Mon 6/4/01	Fri 6/8/01	83	Russ Anderson
85	✓	**3.4**	**Specify NVRAM data structure interface for partition configurations**	**1**	**154 days**	**100%**	**Tue 5/8/01**	**Fri 12/7/01**		**Russ Anderson**
86	✓ 📝	**3.4.1**	**What EFI boot parameters need to be defined? CPUS?, Address ranges?, Memory sizes?**	**1**	**154 days**	**100%**	**Tue 5/8/01**	**Fri 12/7/01**		**Russ Anderson**
87	✓	3.4.1.1	Write Proposal	1	5 days	100%	Tue 5/8/01	Mon 5/14/01		Russ Anderson
88	✓	3.4.1.2	Identify what can be stored in NVRAM that is needed by Linux for partitioning	1	149 days	100%	Tue 5/15/01	Fri 12/7/01	87	Russ Anderson
89	📝	**3.5**	**System management tool to set the partition configuration**	**2**	**5 days**	**0%**	**Mon 7/15/02**	**Fri 7/19/02**		**Russ Anderson**



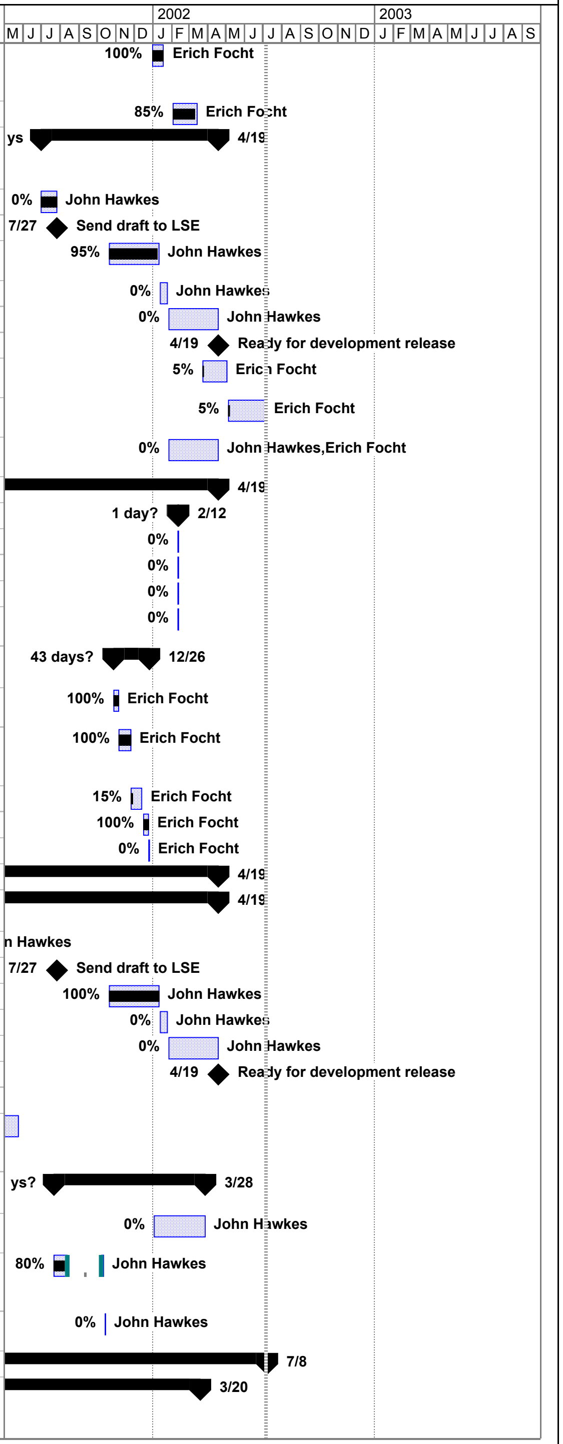
Atlas Task List

ID	ℹ	ID	Task Name	Pri	Duration	% Complet	Start	Finish	Predecessors	Resource Names
90	▦	3.5.1	Write Proposal (outlining how each vendor can write their own interface)	2	5 days	0%	Mon 7/15/02	Fri 7/19/02		Russ Anderson
91		**3.6**	**Licensing issues on partitioned systems**		**236 days**	**29%**	**Mon 11/26/01**	**Mon 10/21/02**		**Russ Anderson**
92	✓	3.6.1	Write Proposal		2 days	100%	Mon 11/26/01	Tue 11/27/01		Russ Anderson
93	▦	3.6.2	Develop patch		5 days	0%	Tue 10/15/02	Mon 10/21/02		Russ Anderson
94		**4**	**Machine Check Handling**		**364.9 days**	**55%**	**Mon 5/21/01**	**Fri 10/11/02**		**Jenna Hall,Al**
95	✓	**4.1**	**Logging**		**166 days**	**100%**	**Mon 5/21/01**	**Mon 1/7/02**		
96	✓	4.1.1	Correctable CPU error reporting and logging support	1	14 days	100%	Mon 5/21/01	Thu 6/7/01		Jenna Hall
97	✓	4.1.2	Memory ECC error reporting and logging support	1	20 days	100%	Fri 6/8/01	Thu 7/5/01	96,18	Jenna Hall
98	✓	4.1.3	Integration - Error injection	1	15 days	100%	Fri 7/6/01	Thu 7/26/01	97	Jenna Hall
99	✓	4.1.4	Testing - Error injection	1	5 days	100%	Tue 1/1/02	Mon 1/7/02	98FS+112	Jenna Hall
100		**4.2**	**OS MCA recovery: -Memory uncorrectable -Cache uncorrectable, IO device**	**2**	**259.9 days**	**49%**	**Mon 10/15/01**	**Fri 10/11/02**		**Zoltán Menyhart**
101	▦	4.2.1	Project definition	2	140 days	75%	Mon 10/15/01	Fri 4/26/02		Zoltán
102		**4.2.2**	**Memory and Others errors**	**2**	**109 days**	**47%**	**Mon 4/29/02**	**Thu 9/26/02**		**Zoltán Menyhart**
103		4.2.2.1	Design	2	40 days	95%	Mon 4/29/02	Fri 6/21/02	101	Zoltán Menyhart
104		4.2.2.2	Implementation	2	30 days	20%	Mon 6/24/02	Fri 8/2/02	103	Zoltán Menyhart
105	▦	4.2.2.3	Tests	2	24 days	0%	Mon 8/26/02	Thu 9/26/02	104	Zoltán Menyhart
106		**4.2.3**	**PCI errors**	**2**	**219.9 days**	**29%**	**Mon 12/10/01**	**Fri 10/11/02**		**Christian**
107		4.2.3.1	Design	2	46 days	75%	Mon 12/10/01	Mon 7/1/02	101	Christian
108		4.2.3.2	Implementation	2	55 days	0%	Mon 7/1/02	Mon 9/16/02	107	Christian
109		4.2.3.3	Test	2	19 days	0%	Mon 9/16/02	Fri 10/11/02	108	Christian
110		**4.2.4**	**Processor errors and physical mode handler features enhancements**	**2**	**50 days**	**28%**	**Mon 6/3/02**	**Fri 8/9/02**		
111	✓	4.2.4.1	Design	2	10 days	100%	Mon 6/3/02	Fri 6/14/02		Jenna Hall
112		4.2.4.2	Implementation	2	20 days	20%	Mon 6/17/02	Fri 7/12/02	111	Jenna Hall
113		4.2.4.3	Test	2	20 days	0%	Mon 7/15/02	Fri 8/9/02	112	Jenna Hall
114		**5**	**cc-NUMA**		**370 days?**	**44%**	**Mon 2/5/01**	**Fri 7/5/02**		**Tony Luck**
115			**cc-NUMA (Basic)**		**365 days?**	**54%**	**Mon 2/12/01**	**Fri 7/5/02**		
116	📝	**5.1**	**Multiple memory lists for memory**	**1**	**261 days**	**63%**	**Mon 2/12/01**	**Mon 2/11/02**	**9,14**	
117	▦	5.1.1	Dig discontig memory development (for 2.4.X)	1	22 days	99%	Mon 2/12/01	Mon 10/22/01		Tony Luck
118	✓	5.1.2	Definition of discontig memory support with interleaving	1	5 days	100%	Mon 4/23/01	Fri 4/27/01	9,14	Tony Luck
119	✓	5.1.3	Implementation of per-node MM list	1	10 days	100%	Mon 4/30/01	Fri 5/11/01	9,14,118	Jack Steiner
120	✓	5.1.4	SRAT table simulation	1	5 days	100%	Wed 5/16/01	Tue 5/22/01		Jenna Hall
121	✓	5.1.5	Simulation of discontig memory with interleaving	1	20 days	100%	Wed 5/23/01	Tue 6/19/01	9,14,120	Tony Luck
122	✓	5.1.6	Integration of ACPI code with NUMA	1	10 days	100%	Mon 9/17/01	Fri 9/28/01	16,121	Tak Kouchi,Jenna
123	▦	5.1.8	Per-node kmalloc	1	60 days	20%	Mon 10/29/01	Fri 2/8/02	9,14,308	Tony Luck
124			Platform testing	1	1 day	0%	Mon 2/11/02	Mon 2/11/02	123	ALL
125	✓ 📝	**5.2**	**Kernel text replication**	**1**	**172 days?**	**100%**	**Mon 4/9/01**	**Tue 12/4/01**		**Tony Luck**
126	✓	5.2.1	Remove dependency on 1-1 mapping for text	1	1 day?	100%	Mon 4/9/01	Mon 4/9/01	9	Tony Luck
127	✓	5.2.2	Copy text into mappable area on other nodes with proper alignment	1	23 days	100%	Tue 4/10/01	Tue 12/4/01	9,126	Tony Luck
128	✓	5.2.3	Starts Aps and provide appropriate ITR mapping	1	10 days	100%	Tue 5/8/01	Mon 5/21/01	9,127	Tony Luck
129	✓	5.2.4	Sanitize _text/_end references	1	10 days	100%	Tue 5/22/01	Mon 6/4/01	9,128	Tony Luck
130	✓	5.2.5	Constant data		10 days	100%	Tue 6/5/01	Mon 6/18/01	129	Tony Luck
131	📝	**5.3**	**Processor node/cell affinity scheduling**	**1**	**265 days?**	**45%**	**Mon 7/2/01**	**Fri 7/5/02**		
132	✓		Node affine extension of original scheduler	1	10 days	100%	Mon 10/15/01	Fri 10/26/01		Erich Focht
133		**5.3.1**	**Multiple run queues**	**1**	**100 days**	**92%**	**Mon 10/29/01**	**Fri 3/15/02**		**Erich Focht**
134	✓	5.3.1.1	Port IBM code (MQ and pool) to IA64 and test it	1	3 days	100%	Mon 10/29/01	Wed 10/31/01		Erich Focht
135	✓	5.3.1.2	extend configurability to allow flexible definition of CPU pools	1	15 days	100%	Thu 11/1/01	Wed 11/21/01	134	Erich Focht
136		5.3.1.3	test CPU-pools scheduler on real NUMA IA64 (AzusA)	1	5 days	50%	Mon 12/10/01	Fri 12/14/01	135	Erich Focht
137	✓	5.3.1.4	Port O(1) MQ scheduler to IA-64		23 days	100%	Mon 1/14/02	Wed 2/13/02		Erich Focht



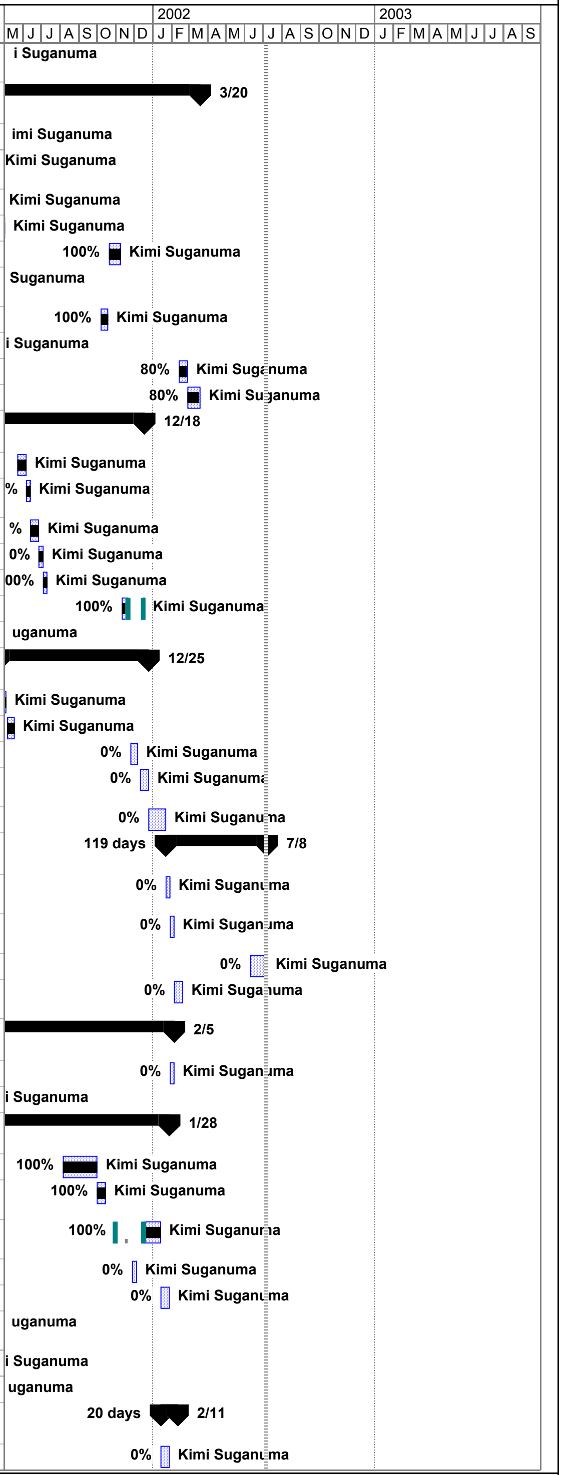
Atlas Task List

ID	ℹ	ID	Task Name	Pri	Duration	% Complet	Start	Finish	Predecessors	Resource Names
138	✓	5.3.1.5	Correct set_cpus_allowed() functionallity which is required by the CpuMemSets and the CpuHotPlug patches		14 days	100%	Tue 1/1/02	Fri 1/18/02		Erich Focht
139		5.3.1.6	Add node-affinity to the O(1) scheduler		30 days	85%	Mon 2/4/02	Fri 3/15/02		Erich Focht
140		**5.3.3**	**Processor/node binding APIs (N.B. 2.4.x kernel already has kernel support for processor binding)**	**1**	**210 days**	**51%**	**Mon 7/2/01**	**Fri 4/19/02**		**John Hawkes**
141	✓	5.3.3.1	Design draft		20 days	100%	Mon 7/2/01	Fri 7/27/01		John Hawkes
142	✓	5.3.3.2	Send draft to LSE		0 days	100%	Fri 7/27/01	Fri 7/27/01	141	John Hawkes
143	▦	5.3.3.3	Implementation (worked on by Paul Jackson)		60 days	95%	Mon 10/22/01	Fri 1/11/02	142FS+30 days	John Hawkes
144		5.3.3.4	Wait for community feedback		10 days	0%	Mon 1/14/02	Fri 1/25/02	143	John Hawkes
145		5.3.3.5	Additional development		60 days	0%	Mon 1/28/02	Fri 4/19/02	144	John Hawkes
146		5.3.3.6	Ready for development release		0 days	0%	Fri 4/19/02	Fri 4/19/02	145	John Hawkes
147	▦	5.3.4	Collect useful information for the scheduler to make run-time decisions relevant to cc-NUMA	1	30 days?	5%	Mon 3/25/02	Fri 5/3/02		Erich Focht
148	▦	5.3.5	Heuristic load balancing scheduler (3 implementation, 3 weeks each?)	1	45 days?	5%	Mon 5/6/02	Fri 7/5/02	147	Erich Focht
149	▦	5.3.6	Alternate default and fall-back memory allocation policies	1	60 days	0%	Mon 1/28/02	Fri 4/19/02		John Hawkes,Erich
150			**cc-NUMA (Advanced)**		**315 days?**	**29%**	**Mon 2/5/01**	**Fri 4/19/02**		
151		**5.4**	**Implementation of platform topology description**	**2**	**1 day?**	**0%**	**Tue 2/12/02**	**Tue 2/12/02**	**116**	
152		5.4.1	Investigate topology data structure	2	1 day?	0%	Tue 2/12/02	Tue 2/12/02	116	
153		5.4.2	Define kernel API for obtaining topology information	2	1 day?	0%	Tue 2/12/02	Tue 2/12/02	116	
154		5.4.3	I/O node proximity utilization	2	1 day?	0%	Tue 2/12/02	Tue 2/12/02	116	
155		5.4.4	Bind device drivers to a Node (beyond what is in 5.5, what does this mean?)	2	1 day?	0%	Tue 2/12/02	Tue 2/12/02	116	
156		**5.5**	**Interrupt affinity; -API to bind an interrupt (support load balancing of interrupts)**	**2**	**43 days?**	**70%**	**Mon 10/29/01**	**Wed 12/26/01**		**Erich Focht**
157	✓	5.5.1	Investigate current status of interrupt affinity implementation	2	7 days	100%	Mon 10/29/01	Tue 11/6/01		Erich Focht
158	✓	5.5.2	Investigate XTPR priority handling improvements and ways to load-balance interrupts	2	14 days	100%	Wed 11/7/01	Mon 11/26/01	157	Erich Focht
159		5.5.3	Improve initial interrupt assignment scheme	2	14 days	15%	Tue 11/27/01	Fri 12/14/01	158	Erich Focht
160	✓	5.5.4	Extend API to new features?	2	7 days	100%	Mon 12/17/01	Tue 12/25/01		Erich Focht
161		5.5.5	Find meaningful benchmark	2	1 day?	0%	Wed 12/26/01	Wed 12/26/01	160	Erich Focht
162		**5.7**	**cc-NUMA specific APIs for applications.**	**1**	**315 days?**	**25%**	**Mon 2/5/01**	**Fri 4/19/02**		
163		**5.7.1**	**Study other public APIs to find which cc-NUMA APIs have already been defined.**	**1**	**315 days**	**40%**	**Mon 2/5/01**	**Fri 4/19/02**		**John Hawkes**
164		5.7.1.1	Framework for invocation	1	20 days	0%	Mon 2/5/01	Fri 3/2/01		John Hawkes
165	▦	5.7.1.2	Send draft to LSE		0 days	0%	Fri 7/27/01	Fri 7/27/01	164	John Hawkes
166	✓	5.7.1.3	Implementation (worked on by Paul Jackson)		60 days	100%	Mon 10/22/01	Fri 1/11/02	165	John Hawkes
167	▦	5.7.1.4	Wait for community feedback		10 days	0%	Mon 1/14/02	Fri 1/25/02	166	John Hawkes
168	▦	5.7.1.5	Additional development		60 days	0%	Mon 1/28/02	Fri 4/19/02	167	John Hawkes
169	▦	5.7.1.6	Ready for development release		0 days	0%	Fri 4/19/02	Fri 4/19/02	168	John Hawkes
170		5.7.2	Glibc entry points.	1	10 days?	0%	Mon 2/5/01	Fri 2/16/01		
171		5.7.3	"Placement" tool to control existing binary applications. [fork/exec hook (2), I/O placement (4)]	1	80 days?	0%	Mon 2/5/01	Fri 5/25/01		
172		**5.9**	**Performance optimization and improvement of kernel due to cc-NUMA.**		**179 days?**	**20%**	**Mon 7/23/01**	**Thu 3/28/02**		
173	▦	5.9.1	Performance guide for cc-NUMA platforms. (ONGOING TASK)		60 days?	0%	Fri 1/4/02	Thu 3/28/02		John Hawkes
174	▦	5.9.2	Need to define success criteria (e.g., comparative performance to other cc-NUMA goals)		20 days	80%	Mon 7/23/01	Fri 10/12/01		John Hawkes
175		5.9.3	Kernel lock granularity/locality (ONGOING TASK)		1 day?	0%	Mon 10/15/01	Mon 10/15/01	174	John Hawkes
176		**6**	**Hot Plug Processor and Memory Support**		**372 days?**	**54%**	**Fri 2/2/01**	**Mon 7/8/02**		**Kimi Suganuma**
177		**6.1**	**Online/Offline processor support in OS; -Define resource re-allocation policy (bound to a CPU) -Dependency on bootstrap processor.**	**1**	**293 days?**	**78%**	**Mon 2/5/01**	**Wed 3/20/02**		**Kimi Suganuma**



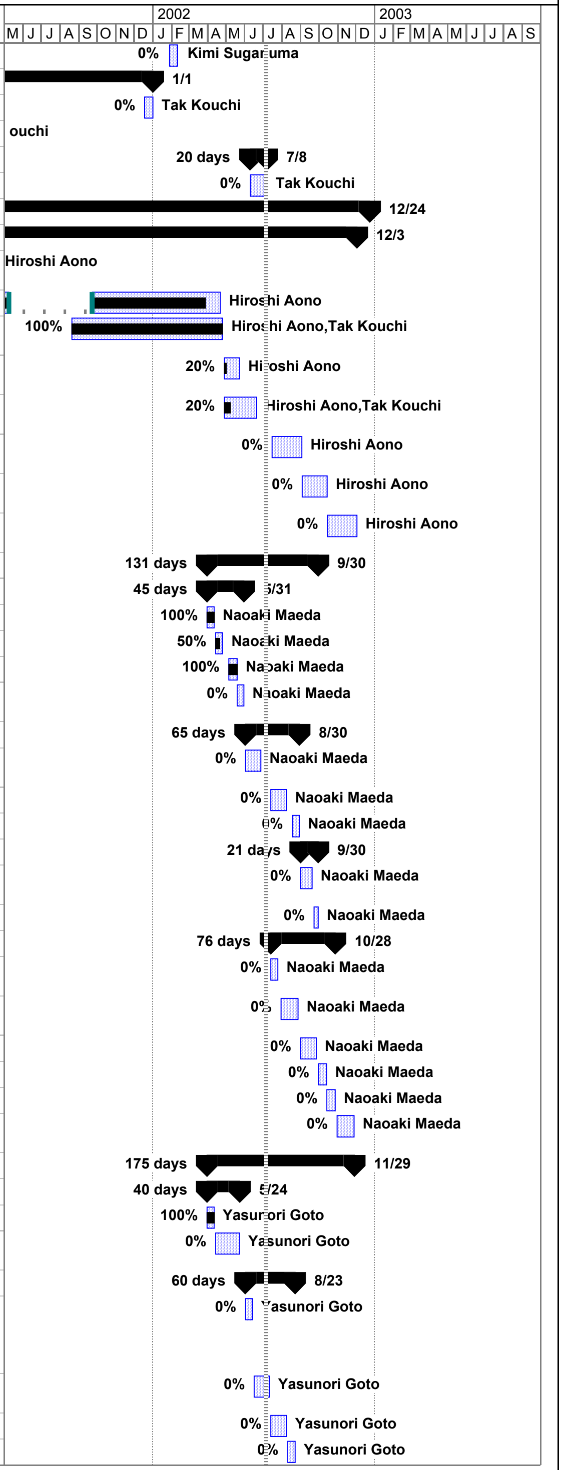
Atlas Task List

ID	ℹ	ID	Task Name	Pri	Duration	% Complet	Start	Finish	Predecessors	Resource Names
178	✓	6.1.1	Investigation for dependency of initial CPU number.	1	10 days	100%	Thu 3/1/01	Wed 3/14/01		Kimi Suganuma
179		**6.1.2**	**Offline processor - Design, Implementation and Function tests**	**1**	**293 days?**	**95%**	**Mon 2/5/01**	**Wed 3/20/02**		**Kimi Suganuma**
180	✓	6.1.2.1	CPU offline sequence.	1	10 days	100%	Thu 3/29/01	Wed 4/11/01	178	Kimi Suganuma
181	✓	6.1.2.2	Remove running process from CPU to be offline.	1	5 days	100%	Thu 4/12/01	Wed 4/18/01		Kimi Suganuma
182	✓	6.1.2.3	Stop interrupts against CPU to be offline.		5 days	100%	Thu 4/19/01	Wed 4/25/01		Kimi Suganuma
183	✓	6.1.2.4	Syncronization and exclusive access control.		5 days	100%	Thu 4/26/01	Wed 5/2/01		Kimi Suganuma
184	✓	6.1.2.5	Stop and reschedule softirq.		15 days	100%	Mon 10/22/01	Fri 11/9/01		Kimi Suganuma
185	✓ ✎	6.1.2.6	Reschedule processes bound to CPU to be offline.		15 days?	100%	Mon 2/5/01	Fri 2/23/01		Kimi Suganuma
186	✓ ✎	6.1.2.7	Investigation for CPU HALT.		10 days	100%	Mon 10/8/01	Fri 10/19/01		Kimi Suganuma
187	✓ ✎	6.1.2.8	Implement and test CPU HALT.		20 days	100%	Mon 2/5/01	Fri 3/2/01		Kimi Suganuma
188	▦ ✎	6.1.2.9	Support Ingo's scheduler		10 days	80%	Thu 2/14/02	Wed 2/27/02		Kimi Suganuma
189	▦	6.1.2.10	Support CpuMemSets		15 days	80%	Thu 2/28/02	Wed 3/20/02		Kimi Suganuma
190		**6.1.3**	**Online processor - Design, Implementation and Function tests**		**227 days?**	**90%**	**Mon 2/5/01**	**Tue 12/18/01**		**Kimi Suganuma**
191	✓	6.1.3.1	CPU online sequence.		10 days	100%	Thu 5/24/01	Wed 6/6/01	178	Kimi Suganuma
192	✓	6.1.3.2	Seek out necessary functions for CPU initialization		5 days	100%	Thu 6/7/01	Wed 6/13/01		Kimi Suganuma
193	✓	6.1.3.3	CPU init function		10 days	100%	Thu 6/14/01	Wed 6/27/01		Kimi Suganuma
194	✓	6.1.3.4	Prepare/Spawn Idle task for CPU to be online		5 days	100%	Thu 6/28/01	Wed 7/4/01		Kimi Suganuma
195	✓	6.1.3.5	Syncronization and exclusive access control		5 days	100%	Thu 7/5/01	Wed 7/11/01		Kimi Suganuma
196	✓	6.1.3.6	Prepare/Spawn ksoftirqd for CPU to be online		10 days	100%	Mon 11/12/01	Tue 12/18/01		Kimi Suganuma
197		6.1.3.7	Get sapicid of CPU to be online		5 days?	0%	Mon 2/5/01	Fri 2/9/01		Kimi Suganuma
198		**6.1.4**	**Interfaces - Design, Implementation and Function tests**	**1**	**177 days**	**50%**	**Mon 4/23/01**	**Tue 12/25/01**		**Kimi Suganuma**
199	✓ ✎	6.1.4.1	Interfaces for operations through /proc	1	10 days	100%	Mon 4/23/01	Fri 5/4/01		Kimi Suganuma
200	✓	6.1.4.2	Interfaces to notify kernel modules	1	10 days	100%	Mon 5/7/01	Fri 5/18/01		Kimi Suganuma
201	▦	6.1.4.3	Interfaces to notify apprication programs	1	10 days	0%	Mon 11/26/01	Fri 12/7/01		Kimi Suganuma
202	▦	6.1.4.4	Interfaces for ACPI (platform dependent modules)	1	10 days	0%	Wed 12/12/01	Tue 12/25/01		Kimi Suganuma
203	▦	6.1.5	System tests for CPU online/offline	1	20 days	0%	Wed 12/26/01	Tue 1/22/02		Kimi Suganuma
204	✎	**6.2**	**Implementation for adding or removing processors/ memory (platform dependent).**	**1**	**119 days**	**0%**	**Wed 1/23/02**	**Mon 7/8/02**		**Kimi Suganuma**
205	▦	6.2.1	Investigation for necessary functions of processor driver	1	5 days	0%	Wed 1/23/02	Tue 1/29/02		Kimi Suganuma
206	▦ ✎	6.2.2	Investigation for necessary functions of memory driver	1	5 days	0%	Wed 1/30/02	Tue 2/5/02		Kimi Suganuma
207	▦	6.2.3	Test processor driver and memory driver	1	20 days	0%	Tue 6/11/02	Mon 7/8/02	27	Kimi Suganuma
208	▦ ✎	6.2.4	Investigation for differences of platform specification	1	10 days	0%	Wed 2/6/02	Tue 2/19/02		Kimi Suganuma
209		**6.3**	**Add/Remove processors in OS; -Dependency on resources that are bound to a Node.**	**1**	**263 days**	**0%**	**Fri 2/2/01**	**Tue 2/5/02**		**Kimi Suganuma**
210	▦	6.3.1	Integration of ACPI with hotplug processor	1	5 days	0%	Wed 1/30/02	Tue 2/5/02	205	Kimi Suganuma
211	▦ ✎	6.3.2	Test using actual hardware	1	20 days	0%	Fri 2/2/01	Thu 3/1/01		Kimi Suganuma
212		**6.4**	**Add memory without a reboot; -VM dependency with boot time memory size.**	**1**	**257 days**	**64%**	**Fri 2/2/01**	**Mon 1/28/02**		**Kimi Suganuma**
213	✓	6.4.1	Investigate for impacts of hot-add memory	1	40 days	100%	Tue 8/7/01	Mon 10/1/01		Kimi Suganuma
214	✓ ✎	6.4.2	Seek out necessary functions for Memory initialization	1	10 days	100%	Tue 10/2/01	Mon 10/15/01		Kimi Suganuma
215	✓	6.4.3	Implement and test hot-add memory function on DISCONTIGMEM	1	20 days	100%	Thu 11/1/01	Mon 1/14/02		Kimi Suganuma
216	▦	6.4.4	Design and Implement user interface	1	5 days	0%	Thu 11/29/01	Wed 12/5/01	211	Kimi Suganuma
217	▦	6.4.5	Integration of ACPI with hotplug memory	1	10 days	0%	Tue 1/15/02	Mon 1/28/02	215	Kimi Suganuma
218	▦ ✎	6.4.6	Integrate of hotplug processor with hotplug memory	1	5 days	0%	Fri 2/2/01	Thu 2/8/01		Kimi Suganuma
219	▦ ✎	6.4.7	Test using actual hardware	1	20 days	0%	Fri 2/2/01	Thu 3/1/01		Kimi Suganuma
220	▦	6.5	Remove portions of memory without reboot.	3	1 day?	0%	Fri 2/2/01	Fri 2/2/01		Kimi Suganuma
221	✎	**6.6**	**System management interfaces for modifying OS to add/remove processors/memory.**	**1**	**20 days**	**0%**	**Tue 1/15/02**	**Mon 2/11/02**		**Kimi Suganuma**
222	✎	6.6.1	Design interface.	1	10 days	0%	Tue 1/15/02	Mon 1/28/02	215,198	Kimi Suganuma



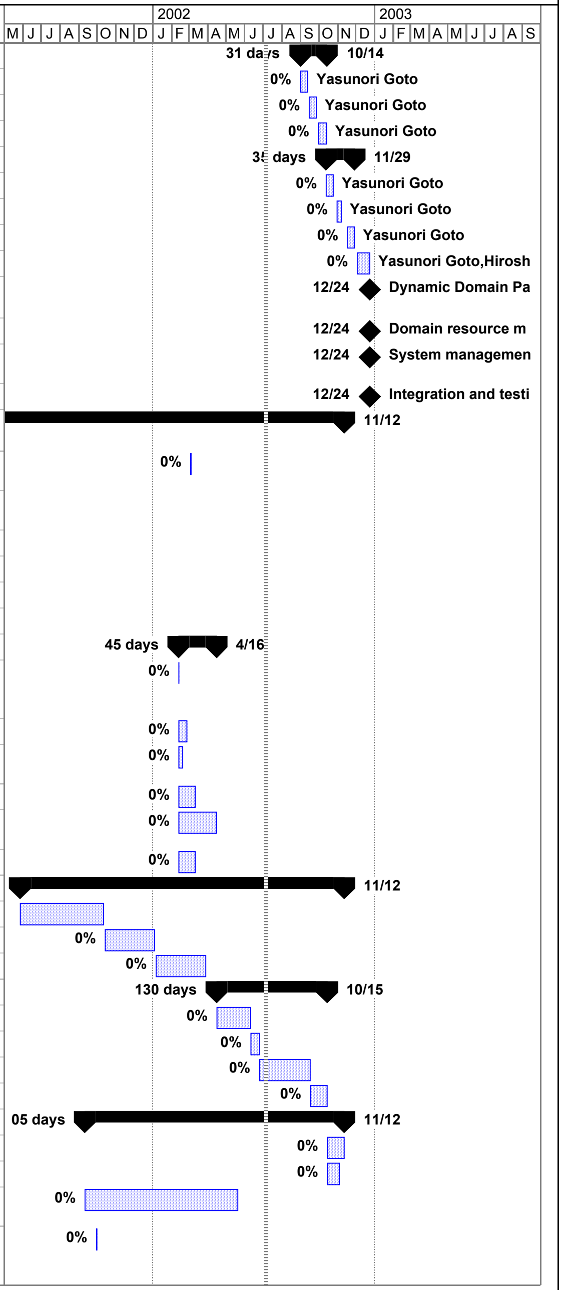
Atlas Task List

ID	ℹ	ID	Task Name	Pri	Duration	% Complet	Start	Finish	Predecessors	Resource Names
223		6.6.2	Implementation interface.	1	10 days	0%	Tue 1/29/02	Mon 2/11/02	222	Kimi Suganuma
224		**6.7**	**System management interfaces document.**	**1**	**237 days**	**0%**	**Mon 2/5/01**	**Tue 1/1/02**		**Kimi Suganuma**
225		6.7.1	Document for Hot Plug user interfaces.	1	10 days	0%	Wed 12/19/01	Tue 1/1/02	190	Tak Kouchi
226		6.7.2	Document of guideline for tools.	1	10 days	0%	Mon 2/5/01	Fri 2/16/01		Tak Kouchi
227		**6.8**	**Update Hot-plug/Hot-remove procedures document.**	**1**	**20 days**	**0%**	**Tue 6/11/02**	**Mon 7/8/02**		**Tak Kouchi**
228		6.8.1	Update document	1	20 days	0%	Tue 6/11/02	Mon 7/8/02	27,193	Tak Kouchi
229		**7**	**Hot Plug IO Node Support**		**449 days**	**44%**	**Thu 4/5/01**	**Tue 12/24/02**		**Hiroshi Aono**
230		**7.1**	**Hot plug PCI device support.**	**1**	**434 days**	**68%**	**Thu 4/5/01**	**Tue 12/3/02**		**Hiroshi Aono**
231	✓	7.1.1	Investigation: current, Compaq, Card-Bus, etc. implementations.	1	10 days	100%	Thu 4/5/01	Wed 4/18/01		Hiroshi Aono
232	📅 📝	7.1.2	Add on or improve current code.	1	166 days	90%	Thu 4/19/01	Mon 4/22/02	231	Hiroshi Aono
233	✓ 📝	7.1.3	ACPI interrupt handler design and implementation. (DEPENDENT ON ACPI 2.0)	1	179 days	100%	Tue 8/21/01	Fri 4/26/02		Hiroshi Aono,Tak Kouchi
234	📅 📝	7.1.4	Function Test Hot-plug PCI with ACPI on an IA-32 machine.	1	20 days	20%	Mon 4/29/02	Fri 5/24/02	232,233	Hiroshi Aono
235	📅 📝	7.1.5	Function Test Hot-plug PCI with ACPI on an IA-64 machine.	1	40 days	20%	Mon 4/29/02	Fri 6/21/02	232,233	Hiroshi Aono,Tak Kouchi
236	📅	7.1.6	Merge Hot Plug PCI device function in IO node management - Design	1	35 days	0%	Wed 7/17/02	Tue 9/3/02	235,265	Hiroshi Aono
237	📅	7.1.7	Merge Hot Plug PCI device function in IO node management - Implementatior		30 days	0%	Wed 9/4/02	Tue 10/15/02	236	Hiroshi Aono
238	📝	7.1.8	Merge Hot Plug PCI device function in IO node management - Function tes		35 days	0%	Wed 10/16/02	Tue 12/3/02	237	Hiroshi Aono
239		**7.2**	**I/O node resource manager for ACPI**	**1**	**131 days**	**23%**	**Mon 4/1/02**	**Mon 9/30/02**		**Hiroshi Aono**
240		**7.2.1**	**Research**	**1**	**45 days**	**63%**	**Mon 4/1/02**	**Fri 5/31/02**		**Hiroshi Aono**
241	✓	7.2.1.1	Study Compaq's PCI Hot-plug implementation.	1	10 days	100%	Mon 4/1/02	Fri 4/12/02		Naoaki Maeda
242	📅	7.2.1.2	Study IBM's PCI Hot-plug implementation.	1	10 days	50%	Mon 4/15/02	Fri 4/26/02		Naoaki Maeda
243	✓	7.2.1.3	Study Linux' current I/O resource management.	1	10 days	100%	Tue 5/7/02	Mon 5/20/02		Naoaki Maeda
244	📅	7.2.1.4	Study the current ACPI PCI Hot-plug implementation.	1	10 days	0%	Mon 5/20/02	Fri 5/31/02	233	Naoaki Maeda
245		**7.2.2**	**Design**		**65 days**	**0%**	**Mon 6/3/02**	**Fri 8/30/02**		
246	📅 📝	7.2.2.1	Preliminary design work and make the design document.		20 days	0%	Mon 6/3/02	Fri 6/28/02	240	Naoaki Maeda
247	📅	7.2.2.2	Design review, discussion and feedback.		20 days	0%	Mon 7/15/02	Fri 8/9/02	246	Naoaki Maeda
248	📅	7.2.2.3	Revise the design document.		10 days	0%	Mon 8/19/02	Fri 8/30/02	247	Naoaki Maeda
249		**7.2.3**	**Implementation**		**21 days**	**0%**	**Mon 9/2/02**	**Mon 9/30/02**		
250	📅 📝	7.2.3.1	Fundamental parts of I/O node resource manager for ACPI.		15 days	0%	Mon 9/2/02	Fri 9/20/02	248	Naoaki Maeda
251	📅 📝	7.2.3.2	Object registration I/F for self debugging.		5 days	0%	Tue 9/24/02	Mon 9/30/02	248	Naoaki Maeda
252		**7.3**	**Hot-plug file system extentions**		**76 days**	**0%**	**Mon 7/15/02**	**Mon 10/28/02**		
253	📅	7.3.1	Study the current PCI hot-plug file system implementation.		10 days	0%	Mon 7/15/02	Fri 7/26/02		Naoaki Maeda
254	📅	7.3.2	Preliminary design work and make the design document.		20 days	0%	Thu 8/1/02	Wed 8/28/02	253	Naoaki Maeda
255	📅	7.3.3	Design review, discussion and feedback.		20 days	0%	Mon 9/2/02	Fri 9/27/02	254	Naoaki Maeda
256	📅	7.3.4	Revise the design document.		10 days	0%	Tue 10/1/02	Mon 10/14/02	255	Naoaki Maeda
257	📅 📝	7.3.5	Implementation		10 days	0%	Tue 10/15/02	Mon 10/28/02	256	Naoaki Maeda
258	📅 📝	7.4	Function Test for I/O node resource manager and hot-plug file system.		20 days	0%	Fri 11/1/02	Thu 11/28/02	251,257	Naoaki Maeda
259		**7.5**	**I/O node driver**		**175 days**	**6%**	**Mon 4/1/02**	**Fri 11/29/02**		
260		**7.5.1**	**Research**		**40 days**	**25%**	**Mon 4/1/02**	**Fri 5/24/02**		
261	✓ 📝	7.5.1.1	Study the ACPI-CA specification		10 days	100%	Mon 4/1/02	Fri 4/12/02		Yasunori Goto
262	📅 📝	7.5.1.2	Prepare for meeting with firmware staffs in Intel to discuss the ACPI specification of Tiger		30 days	0%	Mon 4/15/02	Fri 5/24/02	261	Yasunori Goto
263		**7.5.2**	**Design**		**60 days**	**0%**	**Mon 6/3/02**	**Fri 8/23/02**		
264	📅 📝	7.5.2.1	Discuss the ACPI specification with firmware staffs in Intel (Dependency: and it strongly depends on the schedule of the firmware hot-plug in Tiger.)		10 days	0%	Mon 6/3/02	Fri 6/14/02	262	Yasunori Goto
265	📅	7.5.2.2	Preliminary design work and make the design document.		20 days	0%	Mon 6/17/02	Fri 7/12/02	264	Yasunori Goto
266	📅	7.5.2.3	Design review, discussion and feedback		20 days	0%	Mon 7/15/02	Fri 8/9/02	265	Yasunori Goto
267		7.5.2.4	Revise the design document		10 days	0%	Mon 8/12/02	Fri 8/23/02	266	Yasunori Goto



Atlas Task List

ID	ℹ	ID	Task Name	Pri	Duration	% Complet	Start	Finish	Predecessors	Resource Names
268		**7.5.3**	**Implementation**		**31 days**	**0%**	**Mon 9/2/02**	**Mon 10/14/02**		
269	▦ 📝	7.5.3.1	Initialize and clean-up		10 days	0%	Mon 9/2/02	Fri 9/13/02	267	Yasunori Goto
270	▦ 📝	7.5.3.2	Interrupt handler		10 days	0%	Mon 9/16/02	Fri 9/27/02	267	Yasunori Goto
271	▦ 📝	7.5.3.3	API for upper layer		10 days	0%	Tue 10/1/02	Mon 10/14/02	267	Yasunori Goto
272		**7.5.4**	**Function test of I/O node driver**		**35 days**	**0%**	**Mon 10/14/02**	**Fri 11/29/02**		
273	▦ 📝	7.5.4.1	Initialize and clean-up		10 days	0%	Mon 10/14/02	Fri 10/25/02	269	Yasunori Goto
274	▦ 📝	7.5.4.2	Interrupt handler		5 days	0%	Fri 11/1/02	Thu 11/7/02	270	Yasunori Goto
275	▦ 📝	7.5.4.3	API for upper layer		10 days	0%	Mon 11/18/02	Fri 11/29/02	271	Yasunori Goto
276	▦ 📝	7.6	Integration and testing		15 days	0%	Wed 12/4/02	Tue 12/24/02	230,239,252	Yasunori
277		**8**	**Dynamic Domain Partitioning (contingent upon completion of 6.0, 7.0)**		**0 days**	**0%**	**Tue 12/24/02**	**Tue 12/24/02**		**Asit Mallick**
278		8.1	Domain resource manager.		0 days	0%	Tue 12/24/02	Tue 12/24/02	176,229	
279		8.2	System management interface for hot-plug domain, reset.		0 days	0%	Tue 12/24/02	Tue 12/24/02	278	
280		8.3	Integration and testing.		0 days	0%	Tue 12/24/02	Tue 12/24/02	279	
281		**9**	**Other OS Enhancements (Placeholders- dates not accurate)**		**462 days?**	**0%**	**Mon 2/5/01**	**Tue 11/12/02**		
282	▦	9.1	RAS features; -Dynamic de-allocation of CPU -Dynamic de-allocation of memory page.	3	1 day?	0%	Tue 3/5/02	Tue 3/5/02		
283		9.2	OS Initialization based on ACPI 2.0 including motherboard device enumeration and configuration	3	1 day?	0%	Mon 2/5/01	Mon 2/5/01		
284		9.3	Variable page size support.	3	1 day?	0%	Mon 2/5/01	Mon 2/5/01		
285		9.4	Gang scheduling.	3	1 day?	0%	Mon 2/5/01	Mon 2/5/01		
286		9.5	Check point/restart.	3	1 day?	0%	Mon 2/5/01	Mon 2/5/01		
287		9.6	Batch job/job management.	3	1 day?	0%	Mon 2/5/01	Mon 2/5/01		
288		**9.7**	**Hot page/memory bus load analysis tools**	**3**	**45 days**	**0%**	**Wed 2/13/02**	**Tue 4/16/02**	**151**	
289		9.7.1	Common tools and platform specific register accesses are modularized. Perhaps perfmon extensions to access chipset counters?	3	1 day	0%	Wed 2/13/02	Wed 2/13/02		
290		9.7.2	Detailed list of capabilities of different chipsets	3	10 days	0%	Wed 2/13/02	Tue 2/26/02		
291		9.7.3	Define common denominator. Risk: may be empty set!	3	5 days	0%	Wed 2/13/02	Tue 2/19/02		
292		9.7.4	Generic code for data collection	3	20 days	0%	Wed 2/13/02	Tue 3/12/02		
293		9.7.5	Per-platform code for data collection (3 platforms *3 weeks)	3	45 days	0%	Wed 2/13/02	Tue 4/16/02		
294		9.7.6	User mode analysis & display tools	3	20 days	0%	Wed 2/13/02	Tue 3/12/02		
295		**9.8**	**Page replication/migration.**	**3**	**382 days**	**0%**	**Mon 5/28/01**	**Tue 11/12/02**		
296		9.8.1	Using dynamic feedback provided by platform.	3	100 days	0%	Mon 5/28/01	Fri 10/12/01	171	
297		9.8.2	Scheduler generated process migration.	3	60 days	0%	Mon 10/15/01	Fri 1/4/02	296	
298		9.8.3	Text/read only page replication.	3	60 days	0%	Mon 1/7/02	Fri 3/29/02	297	
299		**9.8.4**	**Replication.**	**3**	**130 days**	**0%**	**Wed 4/17/02**	**Tue 10/15/02**		
300		9.8.4.1	Node-aware page cache.	3	40 days	0%	Wed 4/17/02	Tue 6/11/02	288	
301		9.8.4.2	Optimal pagecopy.	3	10 days	0%	Wed 6/12/02	Tue 6/25/02	300	
302		9.8.4.3	Policy (API, traps, scheduler).	3	60 days	0%	Wed 6/26/02	Tue 9/17/02	301	
303		9.8.4.4	Analysis.	3	20 days	0%	Wed 9/18/02	Tue 10/15/02	302	
304		**9.8.5**	**Migration.**	**3**	**305 days**	**0%**	**Wed 9/12/01**	**Tue 11/12/02**		
305		9.8.5.1	Page fault path.	3	20 days	0%	Wed 10/16/02	Tue 11/12/02	303	
306		9.8.5.2	Integration and testing	3	14 days	0%	Wed 10/16/02	Mon 11/4/02	303	
307	▦	9.8.5.3	Batch and scheduler & command initiated migration.	3	180 days	0%	Wed 9/12/01	Tue 5/21/02		
308		9.9	Memory allocation/deallocation policy (boundary conditions, ongoing) Formerly 5.1.7	1	1 day?	0%	Mon 10/1/01	Mon 10/1/01	9,14,122	


M J J A S O N D J F M A M J J A S O N D J F M A M J J A S
31 days
10/14
0% Yasunori Goto
0% Yasunori Goto
0% Yasunori Goto
35 days
11/29
0% Yasunori Goto
0% Yasunori Goto
0% Yasunori Goto
0% Yasunori Goto,Hirosh
12/24 Dynamic Domain Pa
12/24 Domain resource m
12/24 System managemen
12/24 Integration and testi
11/12
45 days
4/16
11/12
130 days
10/15
05 days
11/12