Team D Project Plan

Cory Kolbeck, Tony Wooster, Erik Swanson, Adrian Miranda, Justin Wagner, and Federico Saldarini

Portland State University
Department of Computer Science
Portland, Oregon

June 10, 2011

Overview and Deliverables

We are to implement client libraries to provide asynchronous communication with a Burrow message queue server.

The code for these libraries will reside on Github, and the sponsor will link to them from the main Burrow site. Documentation will reside on the main Burrow website.

What is Burrow?

"Burrow is a message queue that can be used in a variety of environments, from simple in-process queues to multi-tenant cloud services. The design is extremely modular and can be configured or extended to accommodate many use cases."

(burrow.openstack.org)

Assumptions

- The code will be maintained by the OpenStack community
- The code will continue to reside on Github, or possibly be moved to launchpad
- Documentation will be pushed to Burrow's launchpad `bzr` repo, and hosted on burrow.openstack.org
- Authentication is outside the scope of this project

Restraints

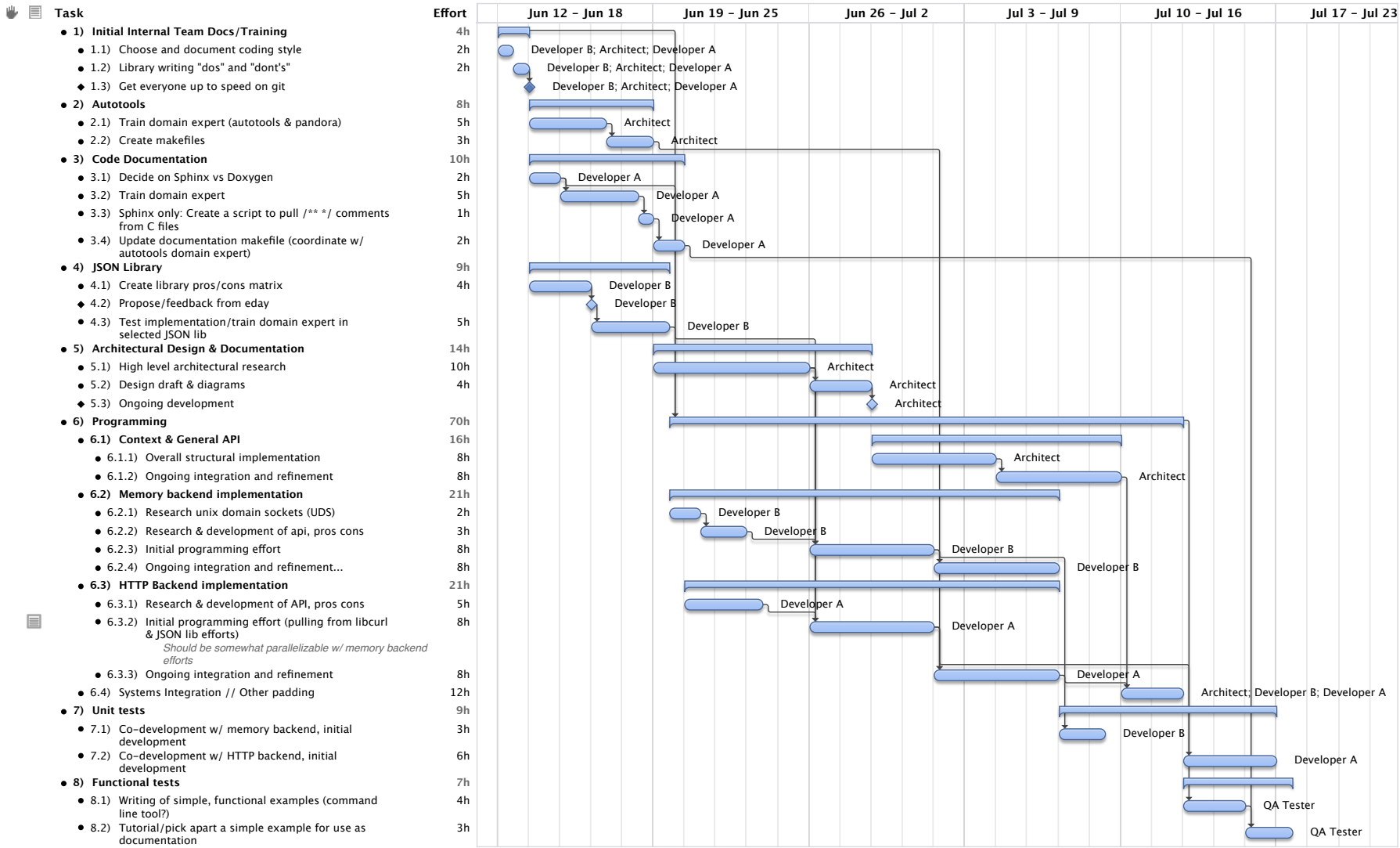
- Project will be distributed under the Apache2 license, and any libraries used must be compatible.
- All calls in to our library must be nonblocking.
- Maven will be used for Java builds.
- The Pandora autoconf macro set will be used for C builds.
- Minimal dependence on external libraries.

Plan Overview

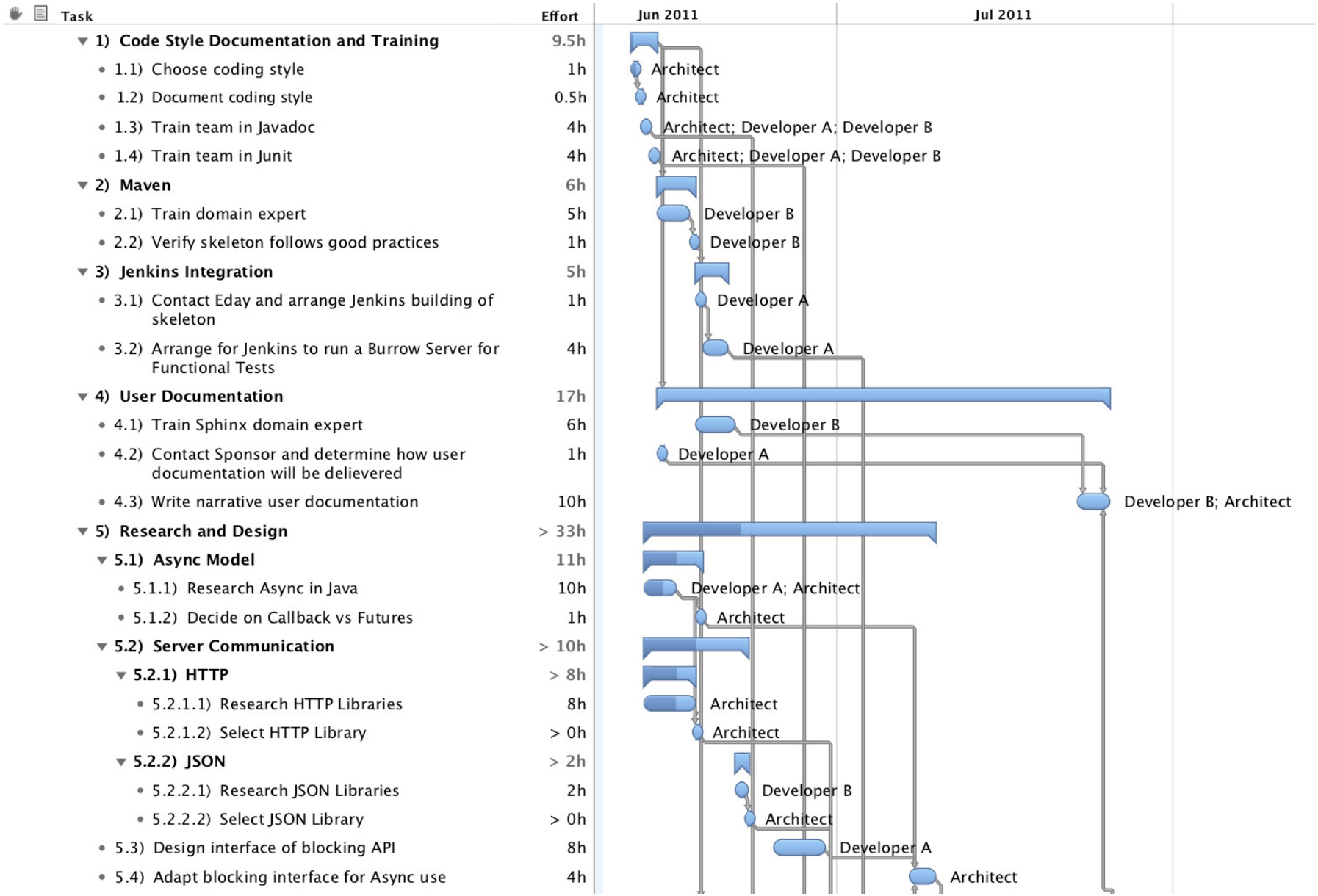
The team will be split into two teams of three. Erik, Justin, and Cory will create a library in Java. Tony, Fede and Adrian will create a library in C.

1. ~~Meet with Eric Day to discuss project details~~
2. ~~Decide on target languages~~
3. ~~Create skeleton projects and set up burrow continuous integration infrastructure~~
4. Research asynchronous I/O
5. Create blocking memory backends
6. Research and choose JSON and HTTP libraries
7. Create blocking http backend
8. ~~Choose language appropriate callback mechanisms~~
9. Write asynchronous memory backend
10. Write asynchronous http backend
11. Write functional tests
12. *Time Allowing* Write small projects which use our libraries in interesting ways.

C Gantt Chart

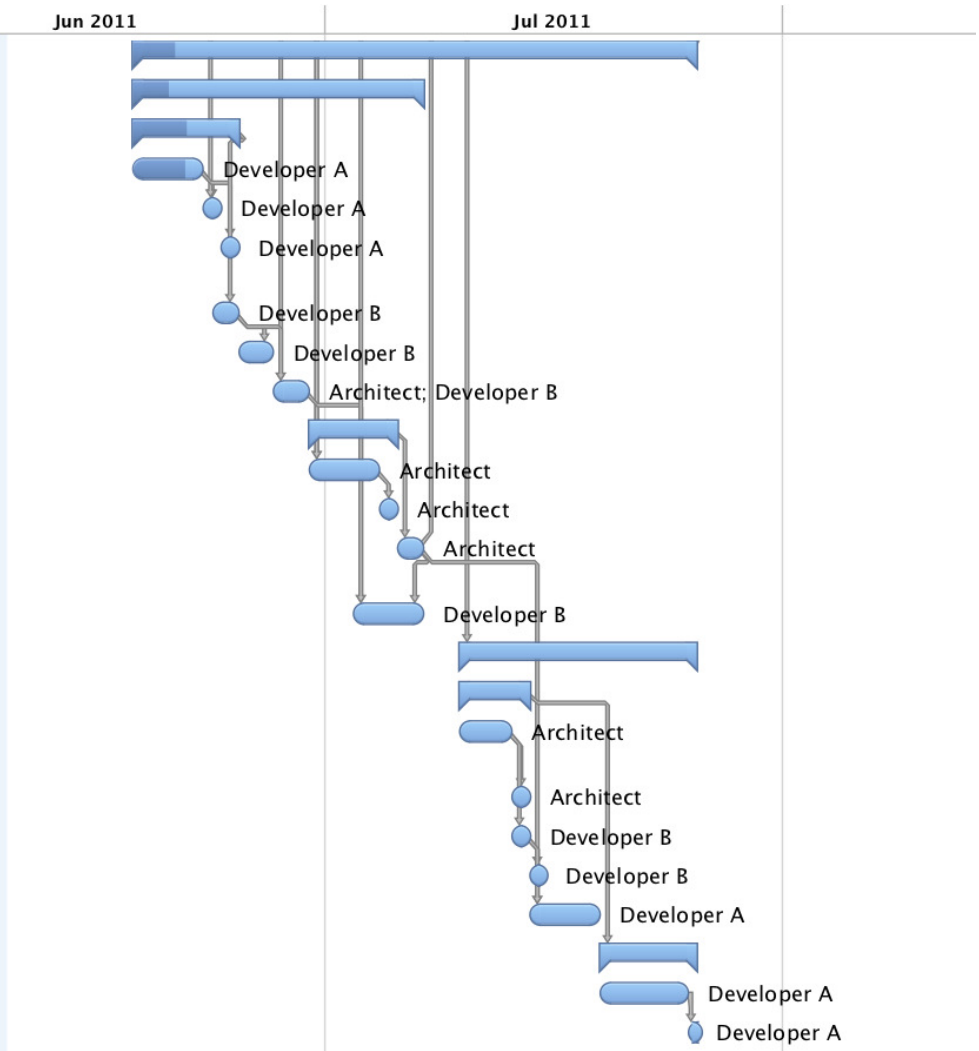


Java Gantt Chart



Java Gantt Chart

Task	Effort
6) Implementation and Code Documentation	79h
6.1) Blocking API	48h
6.1.1) Memory Backend	12h
• 6.1.1.1) Implement Memory Backend	8h
• 6.1.1.2) Document Memory Backend	2h
• 6.1.1.3) Create Direct Unit Tests for Memory Backend	2h
• 6.1.2) Implement Blocking Frontend	3h
• 6.1.3) Document Blocking Frontend	4h
• 6.1.4) Create Blocking Fronend Unit Test Suite	8h
6.1.5) Blocking HTTP Backend	10h
• 6.1.5.1) Implement Blocking HTTP Backend	8h
• 6.1.5.2) Document Blocking HTTP Backend	2h
• 6.1.6) Extend Blocking Frontend to support HTTP	3h
• 6.1.7) Develop Integration Tests	8h
6.2) Async API	31h
6.2.1) Async Memory Backend	8h
• 6.2.1.1) Implement Async Reactor for Memory Backend	6h
• 6.2.1.2) Document Memory Reactor	2h
• 6.2.2) Implement Async Frontend	2h
• 6.2.3) Document Async Frontend	2h
• 6.2.4) Adapt Test Suite to Async Frontend	8h
6.2.5) Async HTTP Backend	11h
• 6.2.5.1) Implement Async HTTP Backend	10h
• 6.2.5.2) Document Async HTTP Backend	1h



C Milestones

Week of	*Deliverable*
Jun 05	Pert Chart, Architecture Overview, Use of Jenkins CI
Jun 12	Arch Discussed/Approved w/ Sponsor, Support Libs, Docs, Source Style chosen
Jun 19	-
Jun 26	Simplified Async HTTP and Memory Backend implemented; Architecture largely documented
Jul 03	HTTP & Memory Component Implementation, First Long-Term Unit Tests
Jul 10	Development & Testing Continues – First True Functional Tests / Demo Programs
Jul 17	Feature Complete w/ Bugs & Testing Gaps, Docbook Begins
Jul 24	.. slack ..
Jul 31	Feature Complete w/ 70% test coverage, finalization: Project Closing & Deliverables
Aug 08	Final Presentation

Java Milestones

Week of	*Deliverable*
Jun 05	Pert Chart, Architecture Overview, Use of Jenkins CI
Jun 12	-
Jun 19	-
Jun 26	Blocking Memory Backend
Jul 03	Blocking HTTP Backend
Jul 10	-
Jul 17	Async Memory Backend
Jul 24	Async HTTP Backend
Jul 31	Narrative Documentation, Sponsor Delivery
Aug 08	Final Presentation

Meetings and Reviews

- In person meetings with sponsor every 1-2 weeks.
- IRC consultation as needed.
- Reviews at milestones as previously noted.

Resource Identification

Name	*Available Hours/Week*
Justin	8-10
Adrian	8-10
Tony	10+
Erik	10-15
Federico	10+
Cory	10-15

Configuration Management

- Github will be used for source control
- Ticketing and bug reporting will be through Github's builtin utilities
- Should it become necessary, language leads will be in charge of resolving merge conflicts
- Unit testing and code coverage reports will be through the Jenkins continuous integration framework.

Roles

Role	*Responsibility*	Initial
Manager	Coordinate general geetings and maintain schedules	C
POC	Maintain communication between team and sponsor	C
Integration	Support Jenkins and Github issues	C
Java Lead	Architect Java library and delegate coding and research tasks	E
C Lead	Architect C library and delegate coding and research tasks	T
Java Dev	Implement designs of Java lead	CJE
C Dev	Implement designs of C lead	AFT
Support	Maintain Shared Machines	C
Unit Testing	Code unit tests for every function	All
Func. Testing	Create functional tests for each language	tbd
API Docs	Write language specific documentation	tbd
General Docs	Create a language agnostic guide to coding burrow clients	T

Risk Management

- Risk: Team member drops out
  Consequence: Fewer man-hours available
  Mitigation: Scheduling a week of slack time
- Risk: Architect drops out
  Consequence: Possible loss of grand plan
  Mitigation: Documentation, regular meetings to keep team members in the loop
- Risk: Sponsor pulls out/disappears
  Consequence: Main link to Burrow project severed, loss of technical guidance
  Mitigation: Forming a relationship with other members of the OpenStack community working on Burrow

QA and Deployment

- Unit and regression testing will be ongoing using Burrow's existing Jenkins continuous integration system.
- Functional testing will take place in the weeks leading up to code freeze.
- Deployment will consist of linking to our existing Github repositories (or possibly official forks) from burrow.openstack.org.
- Documentation will be pushed to the Burrow `bzr` for inclusion in the Burrow wiki.