

Growing OO C++ Software

Guided by Tests



Alan Griffiths

alan@octopull.co.uk

#alanatocpull

Octopull Limited

www.octopull.co.uk

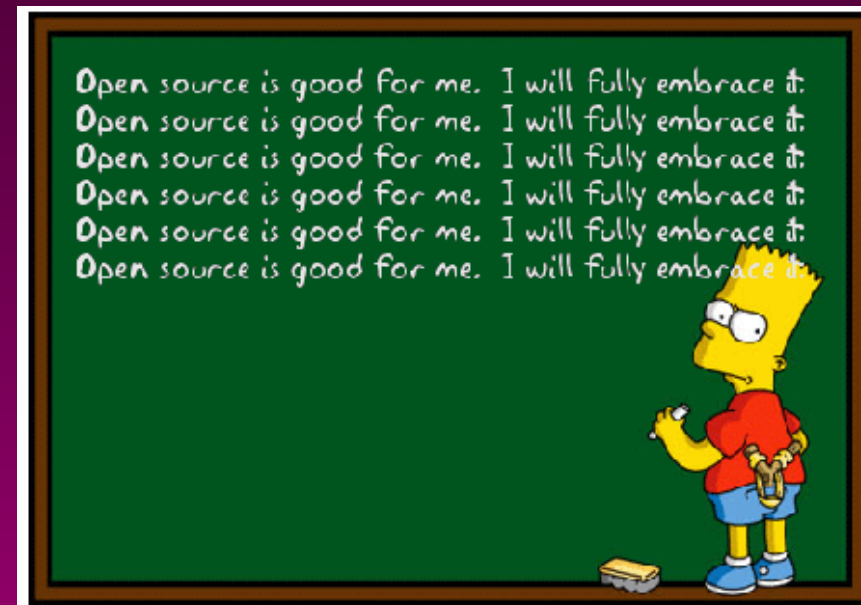
Who am I?

- ◆ **Developed software through many fashions in development processes, technologies and programming languages**
- ◆ **Delivered working software (and development processes)**
- ◆ **A regular at the ACCU conference**
- ◆ **Current ACCU Chair**
- ◆ **Written for magazines and books**
- ◆ **Spoken at a number of conferences**
- ◆ **Made many friends in developer communities**



A project I can talk about

- ◆ Usually clients want confidentiality
 - Canonical doesn't
- ◆ This is an open-source (LGPL/GPL) project
- ◆ Documentation and code is online
 - I've borrowed shamelessly
 - Links at the end



Canonical and me

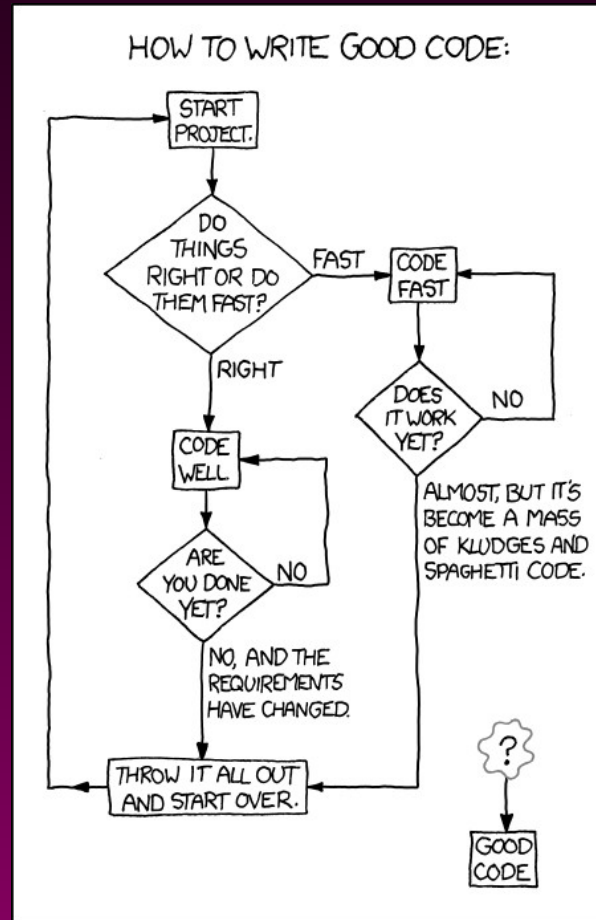
- ◆ **Canonical make Ubuntu Linux**
- ◆ **Their “Unity” user interface**
 - **Is a Compiz plugin**
 - **Designed to cover PC, tablet, phone and TV**
- ◆ **Contracted me to**
 - **Improve quality of Compiz codebase**
 - **Review Compiz and alternative codebases**
- ◆ **Then to work on “something new”**

Rewriting Software

- ◆ The organisation hasn't changed
- ◆ Existing software incorporates many lessons
- ◆ Often a moving target◆ Unless significant
 - Changes in requirements
 - New technology

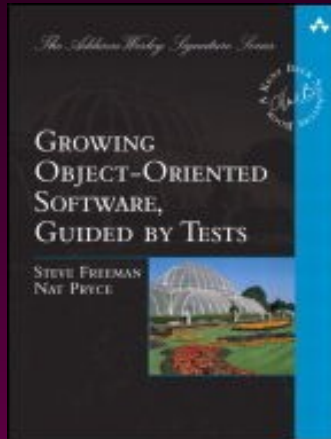


Writing good code



<http://xkcd.com/844/>

Growing Object-Oriented Software, Guided by Tests



By: Steve Freeman & Nat Pryce

<http://www.growing-object-oriented-software.com/>

- ◆ **Required reading for team**
- ◆ **Inspiration for development approach**
- ◆ **A good read**

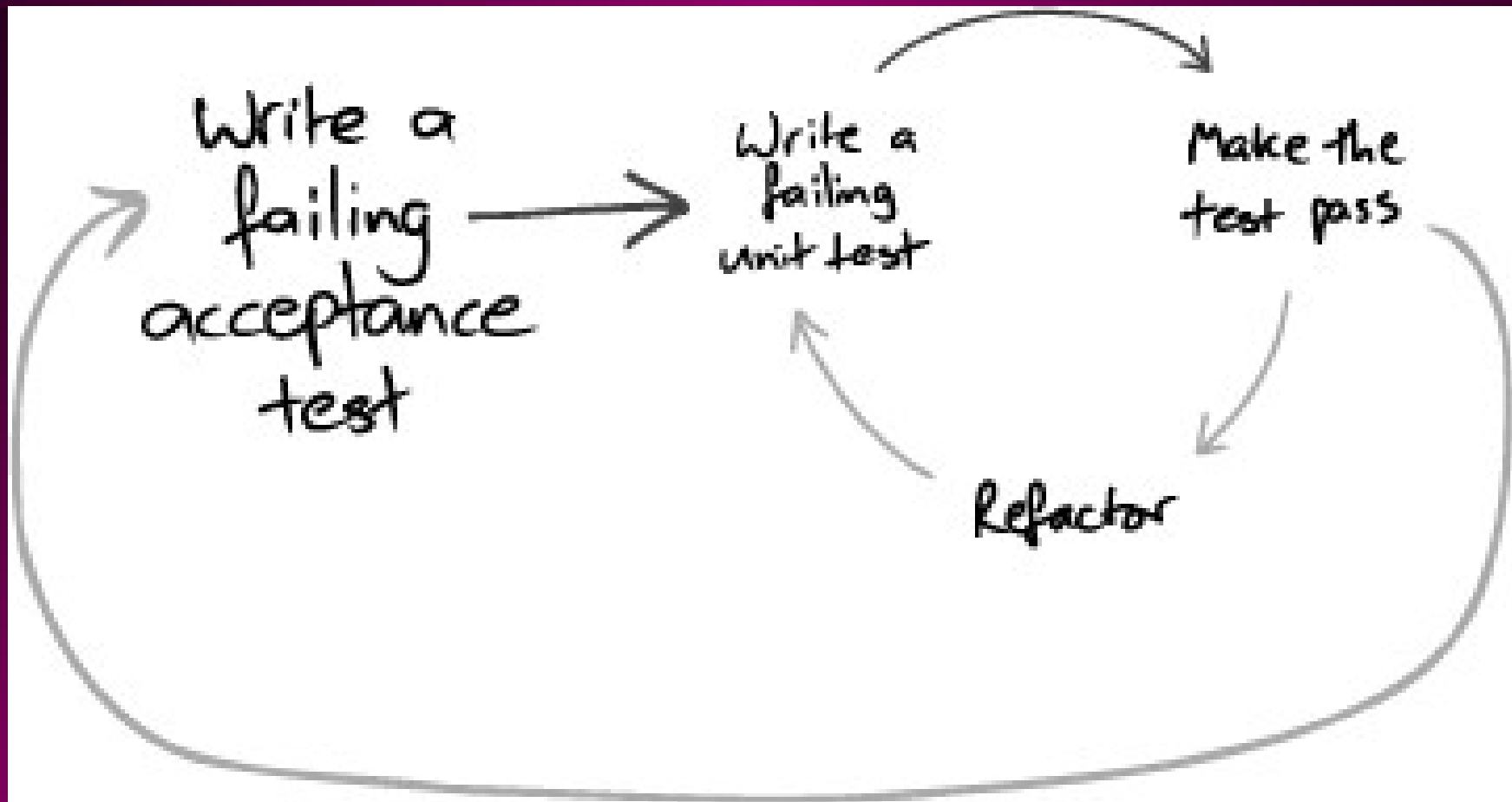
GOOS - Chapter 1

What is the Point of Test-Driven Development?

- ◆ **Software Development as a Learning Process**
- ◆ **Feedback Is the Fundamental Tool**
- ◆ **Practices That Support Change**
- ◆ **Test-Driven Development in a Nutshell**
- ◆ **The Bigger Picture**
- ◆ **Testing End-to-End**
- ◆ **Levels of Testing**
- ◆ **External and Internal Quality**

GOOS - Chapter 1

Inner and outer feedback loops in TDD



GOOS - Chapter 1

Feedback Is the Fundamental Tool

“In a project organized as a set of nested feedback loops, development is incremental and iterative.

“Incremental development builds a system feature by feature, instead of building all the layers and components and integrating them at the end. Each feature is implemented as an end-to-end “slice” through all the relevant parts of the system. The system is always integrated and ready for deployment.

“Iterative development progressively refines the implementation of features in response to feedback until they are good enough.”

“You Can't do TDD in C++”

- ◆ **Chris Matts “challenge” at XtC**
 - **His clients made this assertion**
- ◆ **News to me – I've been doing it for years**
 - **TDD is possible in C++**
 - ➔ (although *all* tooling in C++ is hard)
 - **TDD isn't the answer to all problems**
 - ➔ (but is a big improvement on the usual hack & fix)

Project mir - the goals

- ◆ **Support the Unity shell...**
- ◆ **Across multiple form factors**
 - **Phone, tablet, TV, PC, ...**
- ◆ **Provide driver independence**
 - **Mesa, Nvidia, Android, ...**
- ◆ **Exploit GPU acceleration**
- ◆ **System and Session compositing**
- ◆ **Support for Qt and, possibly, other toolkits**
- ◆ **Support for X applications**

The Project: Mir

a new Window Manager

- ◆ **X-Windows+Compiz/kwin/... is successful but:**
 - Designed with distributed application and server
 - Pre-dates sharing GPUs
 - Lots of legacy baggage
- ◆ **Wayland is great but...**
 - Is a protocol, not an implementation
 - Covers functions Canonical didn't need
- ◆ **Weston (reference implementation of Wayland)**
 - Not optimised for Canonical's needs
 - In development (with lots of other stakeholders)

Multiple platforms

- ◆ **Ubuntu 12.10**

- **g++ 4.6**



- ◆ **Android NDK**

- **g++ 4.4**



- ◆ **Ubuntu 13.x**

- **g++ 4.7**

- ◆ **Android/libhybris**

- **g++ 4.7**

Multiple Locations

- ◆ Nottingham, UK
- ◆ Athens, Greece
- ◆ Bochum, Germany
- ◆ San Diego, CA, USA
- ◆ Carlisle, PA, USA
- ◆ Perth, Australia
- ◆ ...

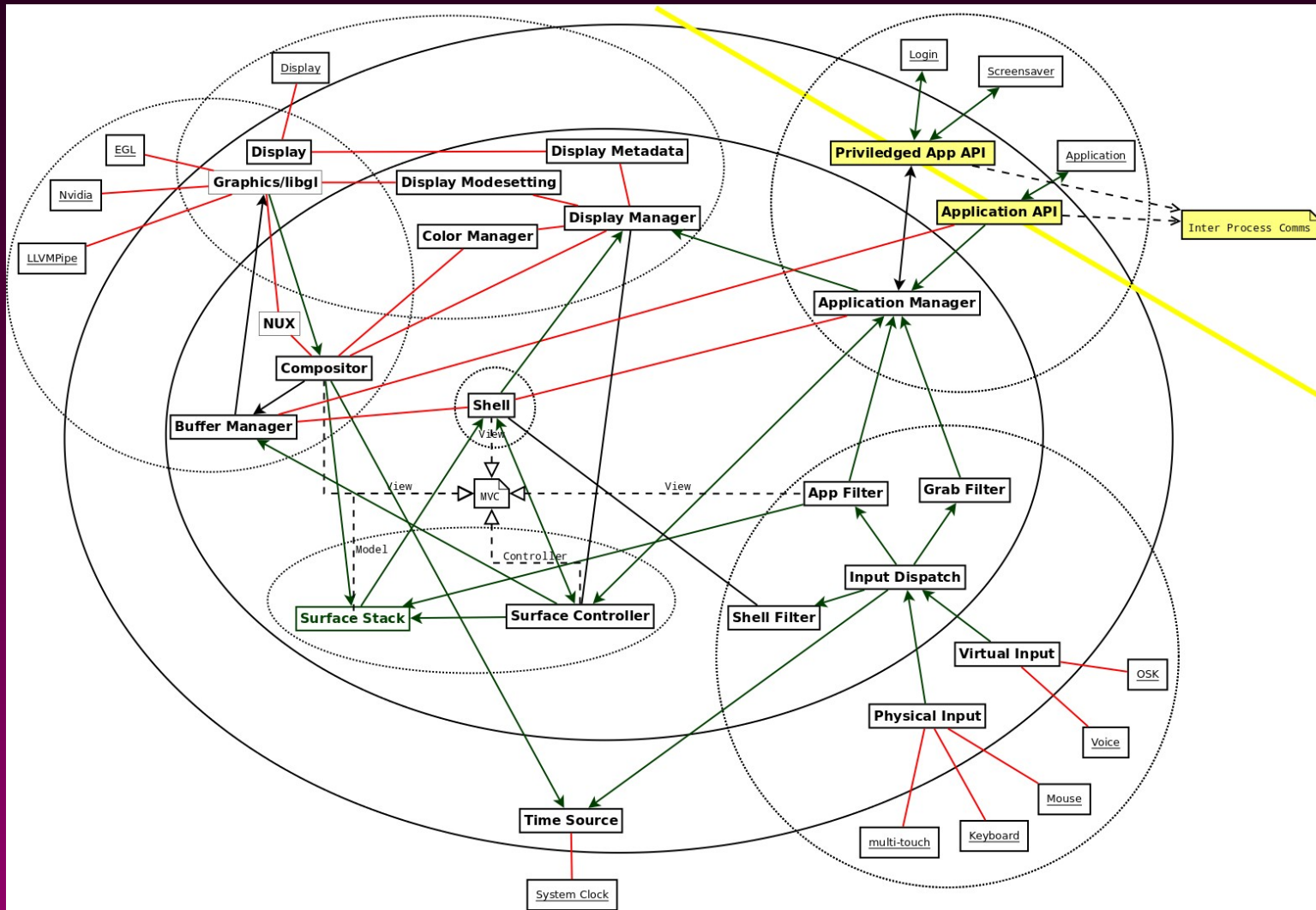
“The most efficient and effective method of conveying information to and within a development team is face-to-face conversation” Agile Manifesto



Forming The Team

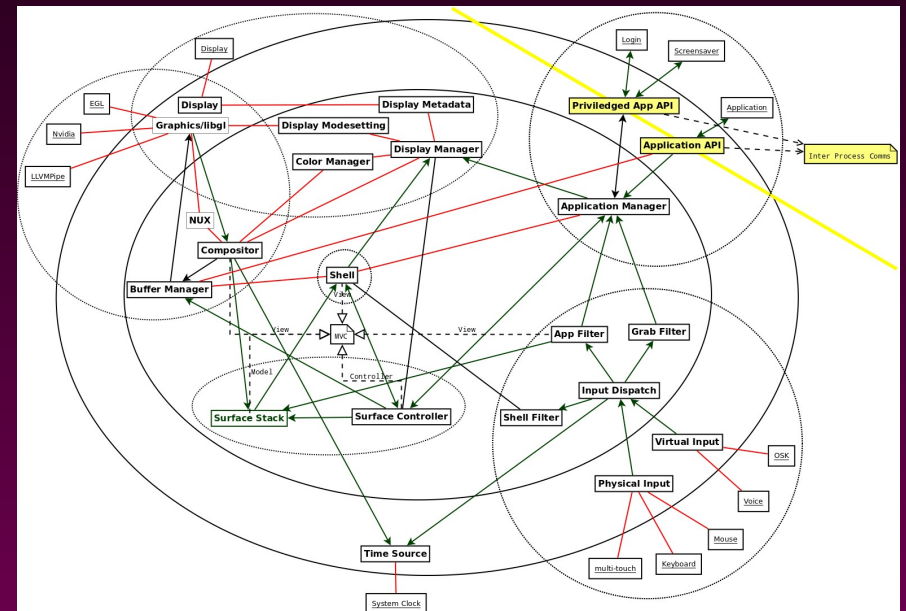
- ◆ **A “Sprint” [Canonical's term]**
 - **Getting the developers together for, typically, a week**
 - **Didn't quite work that way – half the team were in Boston USA and half in London UK**
 - **Agreed development approach – GOOS, C++, CMake, GoogleTest/Mock, Jenkins, bzd**
 - **High level design map**

The High Level Design



Development approach

- Initially worked on the core loop – with “acceptance tests” interfacing inside the outer boundary
- Automating tests for code dealing with real drivers (graphics or input) limited



Dealing with Multiple Locations

- ◆ Standups & meetings
- ◆ Pairing vs code reviews
- ◆ Documentation



Distributed standups & other meetings

- ◆ **Early in the project**
 - **Google Hangouts for standup and discussions**
 - **Skype for one-one**
 - **Email and IRC used, but mostly to arrange above**
- ◆ **Later in the project**
 - **Most discussions on IRC and email**
 - **Some Google Hangouts for other discussion**

Remote pairing

- ◆ **Pair programming at a distance**
 - **TeamViewer**
 - **Different timezones**
 - **Different editors**
 - **Bandwidth**



The friendly All-in-One solution for
**Remote Maintenance, Support,
Remote Access and Home Office**

PDF

Adobe

TeamViewer

The advertisement features a woman with long dark hair, wearing a white shirt, pointing towards a product box. The box is white with blue and green accents and has the TeamViewer logo on it. Below the box is a green badge that says 'All-in-One'. In the bottom left corner, there is a red 'PDF' icon and the Adobe logo. In the bottom right corner, there is the TeamViewer logo.

Code reviews

- ◆ **Code reviews of merge proposals**
 - **Less “bandwidth”**
 - **More “latency”**
 - **Some frustration**
 - **But easier to schedule across timezones**
- ◆ **“Merge Proposals”**
 - **managed by Launchpad website**
 - ➔ **Shows diff**
 - ➔ **Shows comments and review status**
 - **Monitored by Jenkins**
 - ➔ **Builds supported configurations**
 - ➔ **Merges “Approved” changes**

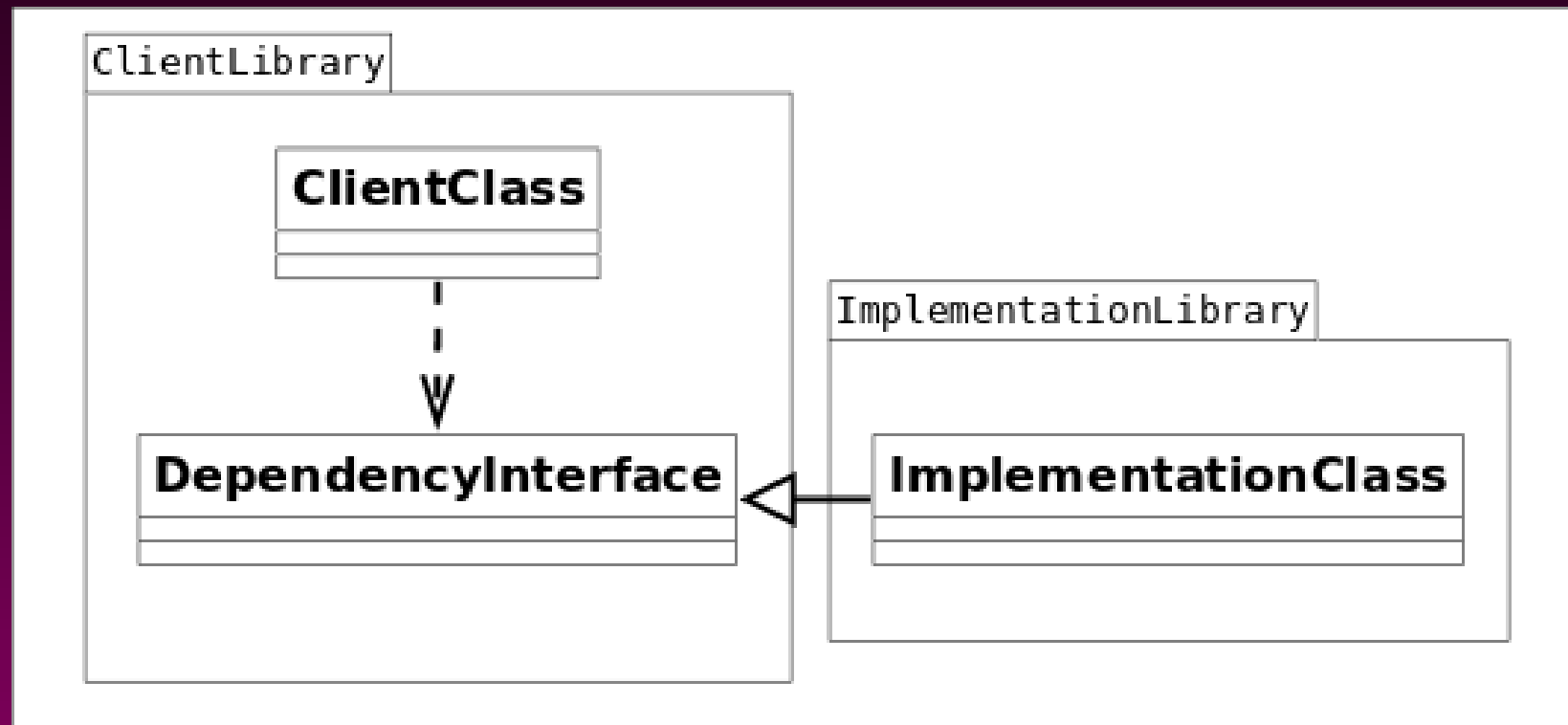
Documentation

- ◆ **Launchpad**
 - **Source control**
 - **Merge Proposals**
 - **Bugs**
 - **Blueprints**
- ◆ **Source code**
 - **Doxygen (code + markdown)**
 - **Design images**
- ◆ **Google docs**
 - **Discussion documents**
 - **Working notes**
 - **Not deliverables/public**
- ◆ **Email**
- ◆ **Wiki**

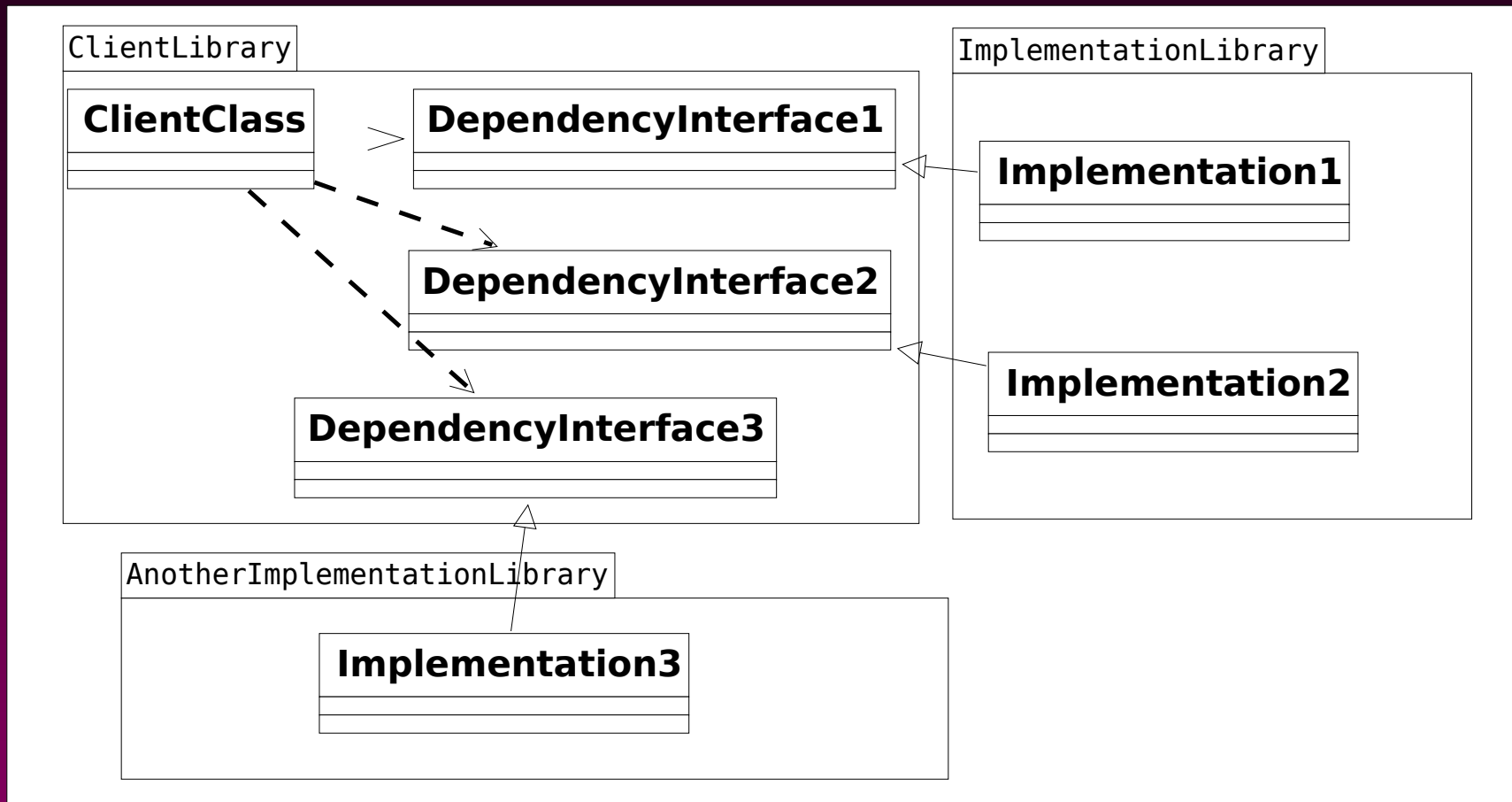
Documentation – in the code

```
// TODO comparing strings in an if-else chain isn't efficient.  
// It is probably possible to generate a Trie at compile time.  
if ("connect" == invocation.method_name())  
{  
    invoke(&protobuf::DisplayServer::connect, invocation);  
}  
else if ("create_surface" == invocation.method_name())  
...  
...
```

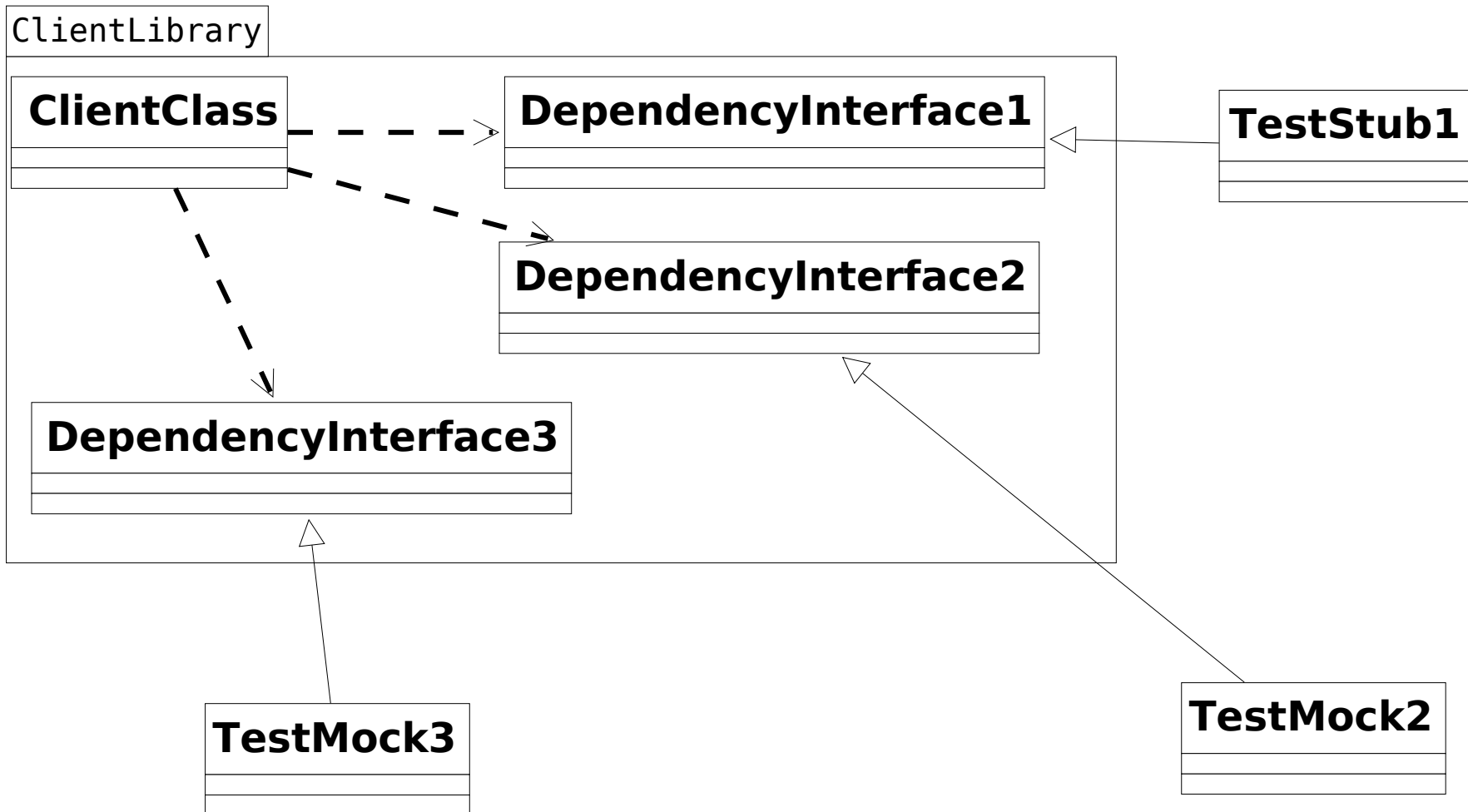
TDD : use interfaces



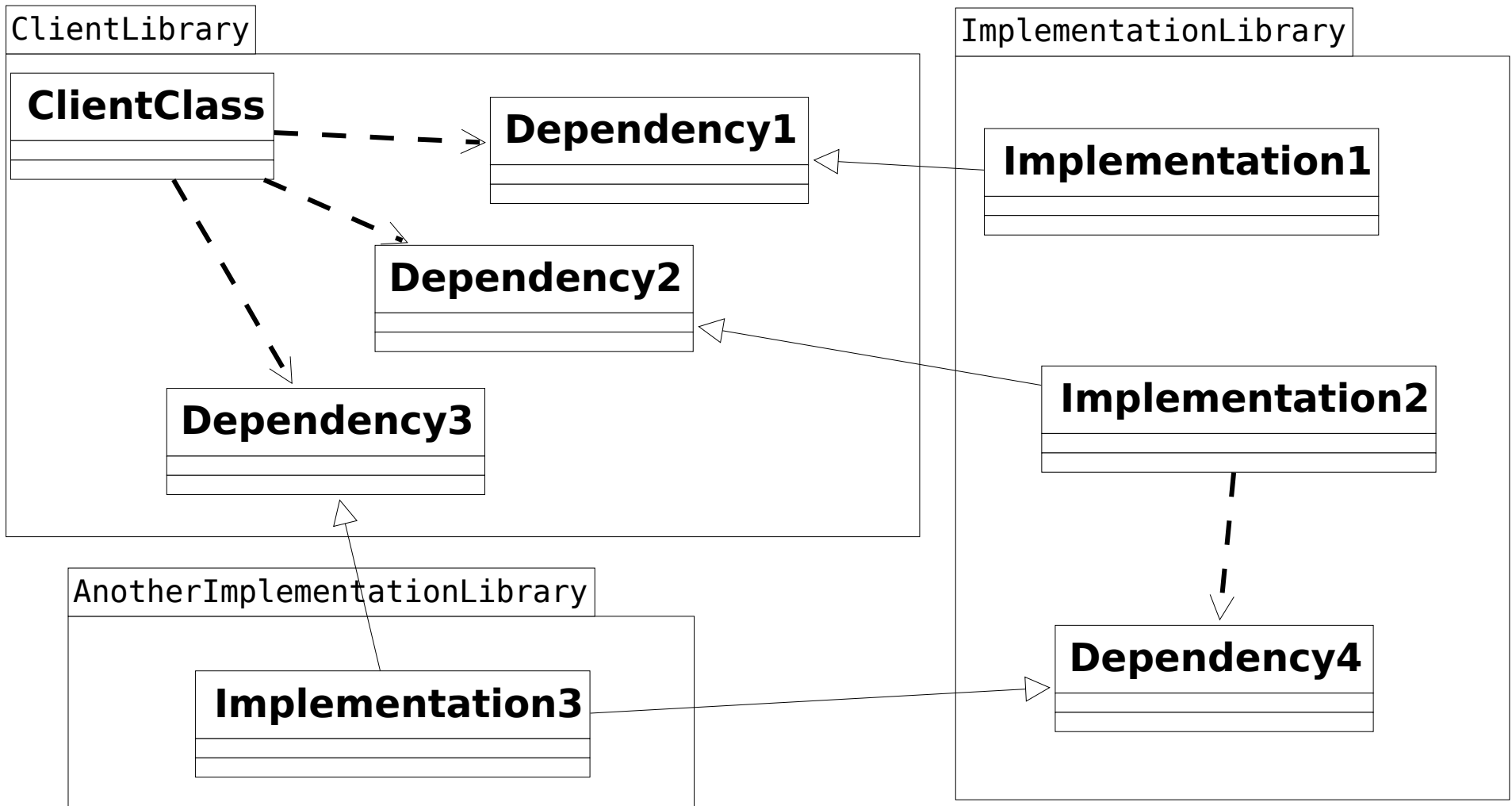
Design using Interfaces



Unit Testing



Design using Interfaces



A unit test

```
struct ShellSurface : testing::Test
{
    std::shared_ptr<mi::InputChannel> const null_input_channel;
    MockSurfaceBuilder surface_builder;
};

TEST_F(ShellSurface, creation_and_destruction)
{
    using namespace testing;

    mf::SurfaceCreationParameters const params;

    InSequence sequence;
    EXPECT_CALL(surface_builder, create_surface(params)).Times(1);
    EXPECT_CALL(surface_builder, destroy_surface(_)).Times(1);

    msh::Surface test(
        mt::fake_shared(surface_builder),
        params,
        null_input_channel);
}
```

A mock object

```
class MockSurfaceBuilder : public msh::SurfaceBuilder
{
public:
    MockSurfaceBuilder()
    {
        using namespace testing;
        ON_CALL(*this, create_surface(_)).
            WillByDefault(Invoke(&self, &StubSurfaceBuilder::create_surface));

        ON_CALL(*this, destroy_surface(_)).
            WillByDefault(Invoke(&self, &StubSurfaceBuilder::destroy_surface));
    }

    MOCK_METHOD1(create_surface, std::weak_ptr<ms::Surface> (const mf::SurfaceCreationParameters&));

    MOCK_METHOD1(destroy_surface, void (std::weak_ptr<ms::Surface> const&));

private:
    StubSurfaceBuilder self;
};
```

A stub object

```
class StubSurfaceBuilder : public msh::SurfaceBuilder
{
public:
    StubSurfaceBuilder() :
        buffer_bundle(new mtd::NullBufferBundle()), dummy_surface()
    {
    }

    std::weak_ptr<ms::Surface> create_surface(mf::SurfaceCreationParameters const&)
    {
        dummy_surface = std::make_shared<ms::Surface>(mf::a_surface().name, buffer_bundle);
        return dummy_surface;
    }

    void destroy_surface(std::weak_ptr<ms::Surface> const&)
    {
        dummy_surface.reset();
    }

private:
    std::shared_ptr<ms::BufferBundle> const buffer_bundle;
    std::shared_ptr<ms::Surface> dummy_surface;
};
```


A modified test

```
TEST_F(ShellSurface, create_throws_means_no_destroy)
{
    using namespace testing;

    mf::SurfaceCreationParameters const params;

    InSequence sequence;
    EXPECT_CALL(surface_builder, create_surface(params)).Times(1)
        .WillOnce(Throw(std::runtime_error(__PRETTY_FUNCTION__)));
    EXPECT_CALL(surface_builder, destroy_surface(_)).Times(Exactly(0));

    EXPECT_THROW({
        msh::Surface test(
            mt::fake_shared(surface_builder),
            params,
            null_input_channel);
    }, std::runtime_error);
}
```

Configuring dependencies

- ◆ **Dependencies are supplied to constructors**
 - **Objects are known by multiple interfaces; but,**
 - **The initial system state has “one of each”**
- ◆ **A DefaultServerConfiguration class**
 - **Caches the created objects**
 - **Knows what to supply for each interface**
- ◆ **A TestingServerConfiguration class overrides**
 - **For integration & acceptance tests**

ServerConfiguration

```
class ServerConfiguration
{
public:
    virtual std::shared_ptr<frontend::Communicator> the_communicator() = 0;
    virtual std::shared_ptr<shell::SessionStore> the_session_store() = 0;
    virtual std::shared_ptr<graphics::Display> the_display() = 0;
    virtual std::shared_ptr<compositor::Drawer> the_drawer() = 0;
    virtual std::shared_ptr<input::InputManager> the_input_manager() = 0;

protected:
    ServerConfiguration() = default;
    virtual ~ServerConfiguration() = default;

    ServerConfiguration(ServerConfiguration const&) = delete;
    ServerConfiguration& operator=(ServerConfiguration const&) = delete;
};
```

main()

```
int main(int argc, char const* argv[])
try
{
    mir::DefaultServerConfiguration config(argc, argv);

    run_mir(config);
    return 0;
}
catch (mir::AbnormalExit const& error)
{
    std::cerr << error.what() << std::endl;
    return 1;
}
catch (std::exception const& error)
{
    std::cerr << "ERROR: " << boost::diagnostic_information(error) << std::endl;
    return 1;
}
```

run_mir()

```
void run_mir(mir::ServerConfiguration& config)
{
    signal(SIGINT, signal_terminate);
    signal(SIGTERM, signal_terminate);
    signal(SIGPIPE, SIG_IGN);

    mir::DisplayServer server(config);

    signal_display_server.store(&server);

    server.run();
}
```

```
std::atomic<mir::DisplayServer*> signal_display_server;

extern "C" void signal_terminate(int)
{
    while (!signal_display_server.load())
        std::this_thread::yield();

    signal_display_server.load()->stop();
}
```

Configuration magic

```
CachedPtr<graphics::DisplayReport> display_report;
```

```
std::shared_ptr<mg::DisplayReport>
mir::DefaultServerConfiguration::the_display_report()
{
    return display_report(
        [this] -> std::shared_ptr<graphics::DisplayReport>
        {
            if (the_options()->get(log_display, false))
            {
                return std::make_shared<ml::DisplayReport>(the_logger());
            }
            else
            {
                return std::make_shared<mg::NullDisplayReport>();
            }
        });
}
```

Project Testing Framework

- ◆ **Test process**
 - Start server process
 - Start client process(es)
 - Wait for client(s) to complete
 - Signal server to exit
 - Wait for the server to exit
 - Validate expectations & assertions
- ◆ **Facilitates**
 - Injecting test doubles into server
 - Injecting test code into server
 - Injecting test code into client(s)

Configuring Server for Tests

```
class TestingServerConfiguration : public DefaultServerConfiguration
{
public:
    ...

    // Code to run in server process
    virtual void exec(DisplayServer* display_server);

    // Code to run in server process after server exits
    virtual void on_exit(DisplayServer* display_server);
    ...

    // We override the_input_manager in the default server configuration
    // to avoid starting and stopping the full android input stack for tests
    // which do not leverage input.
    virtual std::shared_ptr<input::InputManager> the_input_manager();
    ...
};
```


Starting a server with a mock

```
TEST_F(ApplicationMediatorReport, application_connect_called)
{
    struct Server : TestingServerConfiguration
    {
        std::shared_ptr<mf::ApplicationMediatorReport>
        the_application_mediator_report()
        {
            auto result = std::make_shared<MockApplicationMediatorReport>();

            EXPECT_CALL(*result, application_connect_called(testing::_)).
                Times(1);

            return result;
        }
    } server_processing;

    launch_server_process(server_processing);

    ...
}
```

Starting a test client process

```
...
struct Client: TestingClientConfiguration
{
    void exec()
    {
        mt::TestProtobufClient client(mtf::test_socket_file(), rpc_timeout_ms);

        client.connect_parameters.set_application_name(__PRETTY_FUNCTION__);
        EXPECT_CALL(client, connect_done()).Times(testing::AtLeast(0));

        client.display_server.connect(
            0,
            &client.connect_parameters,
            &client.connection,
            google::protobuf::NewCallback(&client, &mt::TestProtobufClient::connect_done));

        client.wait_for_connect_done();
    }
} client_process;

launch_client_process(client_process);
}
```

Retrospective

“At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly.” - Agile Manifesto

Retrospective

Retrospective Dialogue Sheet T1

1. Start here
This is a dialogue sheet. It is designed to promote good conversation. Team members should seat themselves equally around the sheet so each question can be read by at least one person easily. Take one question at a time, skip questions if you like. The person closest to the question should read out the question and take notes of the discussion. Each person should get a chance to read and note at least one question.

2. Set up
Make sure everyone has a pen to write on this sheet.
Agree how long you will spend working on this sheet and write it in this box: 1h

3. Set up
Does everyone agree to follow Ken's Prime Directive (left) while working on this sheet?
The aim of this sheet is to find cause of problems not people.
Leave a space for notes. Use this space however you like notes, ideas, comments and suggestions.

4. Create a timeline
Create a timeline for the period you are considering (e.g. iteration, release, project) in the space above, marking the first and last dates you wish to consider then mark significant and memorable events.
You don't have to agree on everything, discussion is good. The point is to have a good discussion.

5. Successes
What do you consider to be the greatest successes of this work?
Highlight successes on the timeline or write it on the sheet.

6. Difficulties
What difficulties did you encounter during this work?
Record the difficulties on the timeline or elsewhere sheet.
Remember to list the question reader lead the discussion.

7. Keep
Make a list of all the things you did which you want to keep doing.
Make a long list, include all ideas then quickly count (and record) how many of you agree with each idea.

8. Do different
Make a list of things you might do differently to make things better next time.
Write the list on this sheet, include as many ideas as possible - at least 4!

9. Action plan
From the list in #8, choose 3 things you will do, or not do, to make the next piece of work better?

10. Sign-up
Everyone who took part in this exercise, and agrees with the actions should sign here.

Before you begin
The dialogue sheet is intended for use by a team of 2-6 people. If you have more than 6 people, you will need to split into two groups. Each group should have a designated person to read the questions.

The sheet will take at least one hour to complete and may be over two hours.

13. Software © 2012 by Ken Korbett. All rights reserved. Permission is granted to reproduce and distribute this sheet for personal and professional use only. Copying for other purposes is prohibited. For more information see <http://www.dialoguesheets.com>

14. Ken Korbett

15. Ken Korbett

16. Ken Korbett

17. Ken Korbett

18. Ken Korbett

19. Ken Korbett

20. Ken Korbett

21. Ken Korbett

22. Ken Korbett

23. Ken Korbett

24. Ken Korbett

25. Ken Korbett

26. Ken Korbett

27. Ken Korbett

28. Ken Korbett

29. Ken Korbett

30. Ken Korbett

31. Ken Korbett

32. Ken Korbett

33. Ken Korbett

34. Ken Korbett

35. Ken Korbett

36. Ken Korbett

37. Ken Korbett

38. Ken Korbett

39. Ken Korbett

40. Ken Korbett

41. Ken Korbett

42. Ken Korbett

43. Ken Korbett

44. Ken Korbett

45. Ken Korbett

46. Ken Korbett

47. Ken Korbett

48. Ken Korbett

49. Ken Korbett

50. Ken Korbett

51. Ken Korbett

52. Ken Korbett

53. Ken Korbett

54. Ken Korbett

55. Ken Korbett

56. Ken Korbett

57. Ken Korbett

58. Ken Korbett

59. Ken Korbett

60. Ken Korbett

61. Ken Korbett

62. Ken Korbett

63. Ken Korbett

64. Ken Korbett

65. Ken Korbett

66. Ken Korbett

67. Ken Korbett

68. Ken Korbett

69. Ken Korbett

70. Ken Korbett

71. Ken Korbett

72. Ken Korbett

73. Ken Korbett

74. Ken Korbett

75. Ken Korbett

76. Ken Korbett

77. Ken Korbett

78. Ken Korbett

79. Ken Korbett

80. Ken Korbett

81. Ken Korbett

82. Ken Korbett

83. Ken Korbett

84. Ken Korbett

85. Ken Korbett

86. Ken Korbett

87. Ken Korbett

88. Ken Korbett

89. Ken Korbett

90. Ken Korbett

91. Ken Korbett

92. Ken Korbett

93. Ken Korbett

94. Ken Korbett

95. Ken Korbett

96. Ken Korbett

97. Ken Korbett

98. Ken Korbett

99. Ken Korbett

100. Ken Korbett

101. Ken Korbett

102. Ken Korbett

103. Ken Korbett

104. Ken Korbett

105. Ken Korbett

106. Ken Korbett

107. Ken Korbett

108. Ken Korbett

109. Ken Korbett

110. Ken Korbett

111. Ken Korbett

112. Ken Korbett

113. Ken Korbett

114. Ken Korbett

115. Ken Korbett

116. Ken Korbett

117. Ken Korbett

118. Ken Korbett

119. Ken Korbett

120. Ken Korbett

121. Ken Korbett

122. Ken Korbett

123. Ken Korbett

124. Ken Korbett

125. Ken Korbett

126. Ken Korbett

127. Ken Korbett

128. Ken Korbett

129. Ken Korbett

130. Ken Korbett

131. Ken Korbett

132. Ken Korbett

133. Ken Korbett

134. Ken Korbett

135. Ken Korbett

136. Ken Korbett

137. Ken Korbett

138. Ken Korbett

139. Ken Korbett

140. Ken Korbett

141. Ken Korbett

142. Ken Korbett

143. Ken Korbett

144. Ken Korbett

145. Ken Korbett

146. Ken Korbett

147. Ken Korbett

148. Ken Korbett

149. Ken Korbett

150. Ken Korbett

151. Ken Korbett

152. Ken Korbett

153. Ken Korbett

154. Ken Korbett

155. Ken Korbett

156. Ken Korbett

157. Ken Korbett

158. Ken Korbett

159. Ken Korbett

160. Ken Korbett

161. Ken Korbett

162. Ken Korbett

163. Ken Korbett

164. Ken Korbett

165. Ken Korbett

166. Ken Korbett

167. Ken Korbett

168. Ken Korbett

169. Ken Korbett

170. Ken Korbett

171. Ken Korbett

172. Ken Korbett

173. Ken Korbett

174. Ken Korbett

175. Ken Korbett

176. Ken Korbett

177. Ken Korbett

178. Ken Korbett

179. Ken Korbett

180. Ken Korbett

181. Ken Korbett

182. Ken Korbett

183. Ken Korbett

184. Ken Korbett

185. Ken Korbett

186. Ken Korbett

187. Ken Korbett

188. Ken Korbett

189. Ken Korbett

190. Ken Korbett

191. Ken Korbett

192. Ken Korbett

193. Ken Korbett

194. Ken Korbett

195. Ken Korbett

196. Ken Korbett

197. Ken Korbett

198. Ken Korbett

199. Ken Korbett

200. Ken Korbett

201. Ken Korbett

202. Ken Korbett

203. Ken Korbett

204. Ken Korbett

205. Ken Korbett

206. Ken Korbett

207. Ken Korbett

208. Ken Korbett

209. Ken Korbett

210. Ken Korbett

211. Ken Korbett

212. Ken Korbett

213. Ken Korbett

214. Ken Korbett

215. Ken Korbett

216. Ken Korbett

217. Ken Korbett

218. Ken Korbett

219. Ken Korbett

220. Ken Korbett

221. Ken Korbett

222. Ken Korbett

223. Ken Korbett

224. Ken Korbett

225. Ken Korbett

226. Ken Korbett

227. Ken Korbett

228. Ken Korbett

229. Ken Korbett

230. Ken Korbett

231. Ken Korbett

232. Ken Korbett

233. Ken Korbett

234. Ken Korbett

235. Ken Korbett

236. Ken Korbett

237. Ken Korbett

238. Ken Korbett

239. Ken Korbett

240. Ken Korbett

241. Ken Korbett

242. Ken Korbett

243. Ken Korbett

244. Ken Korbett

245. Ken Korbett

246. Ken Korbett

247. Ken Korbett

248. Ken Korbett

249. Ken Korbett

250. Ken Korbett

251. Ken Korbett

252. Ken Korbett

253. Ken Korbett

254. Ken Korbett

255. Ken Korbett

256. Ken Korbett

257. Ken Korbett

258. Ken Korbett

259. Ken Korbett

260. Ken Korbett

261. Ken Korbett

262. Ken Korbett

263. Ken Korbett

264. Ken Korbett

265. Ken Korbett

266. Ken Korbett

267. Ken Korbett

268. Ken Korbett

269. Ken Korbett

270. Ken Korbett

271. Ken Korbett

272. Ken Korbett

273. Ken Korbett

274. Ken Korbett

275. Ken Korbett

276. Ken Korbett

277. Ken Korbett

278. Ken Korbett

279. Ken Korbett

280. Ken Korbett

281. Ken Korbett

282. Ken Korbett

283. Ken Korbett

284. Ken Korbett

285. Ken Korbett

286. Ken Korbett

287. Ken Korbett

288. Ken Korbett

289. Ken Korbett

290. Ken Korbett

291. Ken Korbett

292. Ken Korbett

293. Ken Korbett

294. Ken Korbett

295. Ken Korbett

296. Ken Korbett

297. Ken Korbett

298. Ken Korbett

299. Ken Korbett

300. Ken Korbett

301. Ken Korbett

302. Ken Korbett

303. Ken Korbett

304. Ken Korbett

305. Ken Korbett

306. Ken Korbett

307. Ken Korbett

308. Ken Korbett

309. Ken Korbett

310. Ken Korbett

311. Ken Korbett

312. Ken Korbett

313. Ken Korbett

314. Ken Korbett

315. Ken Korbett

316. Ken Korbett

317. Ken Korbett

318. Ken Korbett

319. Ken Korbett

320. Ken Korbett

321. Ken Korbett

322. Ken Korbett

323. Ken Korbett

324. Ken Korbett

325. Ken Korbett

326. Ken Korbett

327. Ken Korbett

328. Ken Korbett

329. Ken Korbett

330. Ken Korbett

331. Ken Korbett

332. Ken Korbett

333. Ken Korbett

334. Ken Korbett

335. Ken Korbett

336. Ken Korbett

337. Ken Korbett

338. Ken Korbett

339. Ken Korbett

340. Ken Korbett

341. Ken Korbett

342. Ken Korbett

343. Ken Korbett

344. Ken Korbett

345. Ken Korbett

346. Ken Korbett

347. Ken Korbett

348. Ken Korbett

349. Ken Korbett

350. Ken Korbett

351. Ken Korbett

352. Ken Korbett

353. Ken Korbett

354. Ken Korbett

355. Ken Korbett

356. Ken Korbett

357. Ken Korbett

358. Ken Korbett

359. Ken Korbett

360. Ken Korbett

361. Ken Korbett

362. Ken Korbett

363. Ken Korbett

364. Ken Korbett

365. Ken Korbett

366. Ken Korbett

367. Ken Korbett

368. Ken Korbett

369. Ken Korbett

370. Ken Korbett

371. Ken Korbett

372. Ken Korbett

373. Ken Korbett

374. Ken Korbett

375. Ken Korbett

376. Ken Korbett

377. Ken Korbett

378. Ken Korbett

379. Ken Korbett

380. Ken Korbett

381. Ken Korbett

382. Ken Korbett

383. Ken Korbett

384. Ken Korbett

385. Ken Korbett

386. Ken Korbett

387. Ken Korbett

388. Ken Korbett

389. Ken Korbett

390. Ken Korbett

391. Ken Korbett

392. Ken Korbett

393. Ken Korbett

394. Ken Korbett

395. Ken Korbett

396. Ken Korbett

397. Ken Korbett

398. Ken Korbett

399. Ken Korbett

400. Ken Korbett

401. Ken Korbett

402. Ken Korbett

403. Ken Korbett

404. Ken Korbett

405. Ken Korbett

406. Ken Korbett

407. Ken Korbett

408. Ken Korbett

409. Ken Korbett

410. Ken Korbett

411. Ken Korbett

412. Ken Korbett

413. Ken Korbett

414. Ken Korbett

415. Ken Korbett

416. Ken Korbett

417. Ken Korbett

418. Ken Korbett

419. Ken Korbett

420. Ken Korbett

421. Ken Korbett

422. Ken Korbett

423. Ken Korbett

424. Ken Korbett

425. Ken Korbett

426. Ken Korbett

427. Ken Korbett

428. Ken Korbett

429. Ken Korbett

430. Ken Korbett

431. Ken Korbett

432. Ken Korbett

433. Ken Korbett

434. Ken Korbett

435. Ken Korbett

436. Ken Korbett

437. Ken Korbett

438. Ken Korbett

439. Ken Korbett

440. Ken Korbett

441. Ken Korbett

442. Ken Korbett

443. Ken Korbett

444. Ken Korbett

445. Ken Korbett

446. Ken Korbett

447. Ken Korbett

448. Ken Korbett

449. Ken Korbett

450. Ken Korbett

451. Ken Korbett

452. Ken Korbett

453. Ken Korbett

454. Ken Korbett

455. Ken Korbett

456. Ken Korbett

457. Ken Korbett

458. Ken Korbett

459. Ken Korbett

460. Ken Korbett

461. Ken Korbett

462. Ken Korbett

463. Ken Korbett

464. Ken Korbett

465. Ken Korbett

466. Ken Korbett

467. Ken Korbett

468. Ken Korbett

469. Ken Korbett

470. Ken Korbett

471. Ken Korbett

472. Ken Korbett

473. Ken Korbett

474. Ken Korbett

475. Ken Korbett

476. Ken Korbett

477. Ken Korbett

478. Ken Korbett

479. Ken Korbett

480. Ken Korbett

481. Ken Korbett

482. Ken Korbett

483. Ken Korbett

484. Ken Korbett

485. Ken Korbett

486. Ken Korbett

487. Ken Korbett

488. Ken Korbett

489. Ken Korbett

490. Ken Korbett

491. Ken Korbett

492. Ken Korbett

493. Ken Korbett

494. Ken Korbett

495. Ken Korbett

496. Ken Korbett

497. Ken Korbett

498. Ken Korbett

499. Ken Korbett

500. Ken Korbett

501. Ken Korbett

502. Ken Korbett

503. Ken Korbett

504. Ken Korbett

505. Ken Korbett

506. Ken Korbett

507. Ken Korbett

508. Ken Korbett

509. Ken Korbett

510. Ken Korbett

511. Ken Korbett

512. Ken Korbett

513. Ken Korbett

514. Ken Korbett

515. Ken Korbett

516. Ken Korbett

517. Ken Korbett

518. Ken Korbett

519. Ken Korbett

520. Ken Korbett

521. Ken Korbett

522. Ken Korbett

523. Ken Korbett

524. Ken Korbett

525. Ken Korbett

526. Ken Korbett

527. Ken Korbett

528. Ken Korbett

529. Ken Korbett

530. Ken Korbett

531. Ken Korbett

532. Ken Korbett

533. Ken Korbett

534. Ken Korbett

535. Ken Korbett

536. Ken Korbett

537. Ken Korbett

538. Ken Korbett

539. Ken Korbett

540. Ken Korbett

541. Ken Korbett

542. Ken Korbett

543. Ken Korbett

544. Ken Korbett

545. Ken Korbett

546. Ken Korbett

547. Ken Korbett

548. Ken Korbett

549. Ken Korbett

550. Ken Korbett

551. Ken Korbett

552. Ken Korbett

553. Ken Korbett

554. Ken Korbett

555. Ken Korbett

556. Ken Korbett

557. Ken Korbett

558. Ken Korbett

559. Ken Korbett

560. Ken Korbett

561. Ken Korbett

562. Ken Korbett

563. Ken Korbett

564. Ken Korbett

565. Ken Korbett

566. Ken Korbett

567. Ken Korbett

568. Ken Korbett

569. Ken Korbett

570. Ken Korbett

571. Ken Korbett

572. Ken Korbett

573. Ken Korbett

574. Ken Korbett

575. Ken Korbett

576. Ken Korbett

577. Ken Korbett

578. Ken Korbett

579. Ken Korbett

580. Ken Korbett

581. Ken Korbett

582. Ken Korbett

583. Ken Korbett

584. Ken Korbett

585. Ken Korbett

586. Ken Korbett

587. Ken Korbett

588. Ken Korbett

589. Ken Korbett

590. Ken Korbett

591. Ken Korbett

592. Ken Korbett

593. Ken Korbett

594. Ken Korbett

595. Ken Korbett

596. Ken Korbett

597. Ken Korbett

598. Ken Korbett

599. Ken Korbett

600. Ken Korbett

601. Ken Korbett

602. Ken Korbett

603. Ken Korbett

604. Ken Korbett

605. Ken Korbett

606. Ken Korbett

607. Ken Korbett

608. Ken Korbett

609. Ken Korbett

610. Ken Korbett

611. Ken Korbett

612. Ken Korbett

613. Ken Korbett

614. Ken Korbett

615. Ken Korbett

616. Ken Korbett

617. Ken Korbett

618. Ken Korbett

619. Ken Korbett

620. Ken Korbett

621. Ken Korbett

622. Ken Korbett

623. Ken Korbett

624. Ken Korbett

625. Ken Korbett

626. Ken Korbett

627. Ken Korbett

628. Ken Korbett

629. Ken Korbett

630. Ken Korbett

631. Ken Korbett

632. Ken Korbett

633. Ken Korbett

634. Ken Korbett

635. Ken Korbett

636. Ken Korbett

637. Ken Korbett

638. Ken Korbett

639. Ken Korbett

640. Ken Korbett

641. Ken Korbett

642. Ken Korbett

643. Ken Korbett

644. Ken Korbett

645. Ken Korbett

646. Ken Korbett

647. Ken Korbett

648. Ken Korbett

649. Ken Korbett

650. Ken Korbett

651. Ken Korbett

652. Ken Korbett

653. Ken Korbett

654. Ken Korbett

655. Ken Korbett

656. Ken Korbett

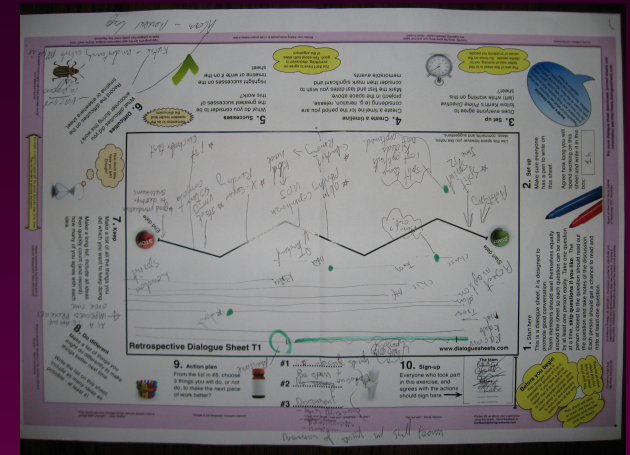
657. Ken Korbett

658. Ken Korbett

659. Ken Korb

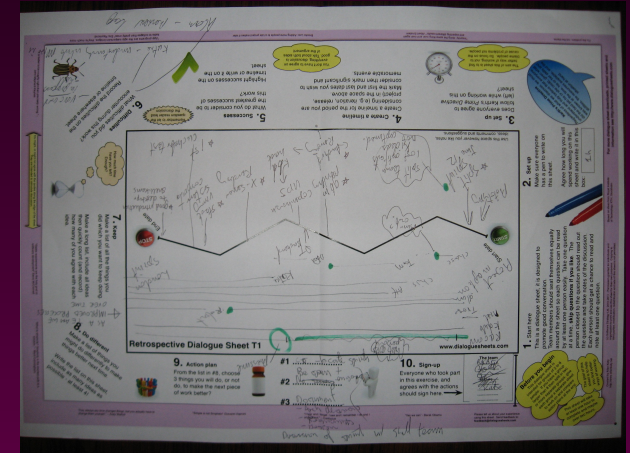
Retrospective

- ◆ **Difficulties**
 - **feedback lag in code reviews**
 - **Understanding what mir is**
 - **Having an “in process” shell**



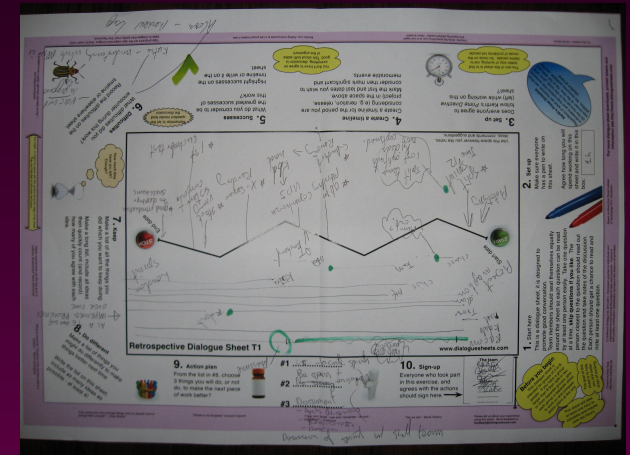
Retrospective

- ◆ **Keep**
 - **Tests**
 - **Code reviews**
 - **Power to veto**
 - **Maturity of discussion & receptiveness to criticism**
 - **Hangouts**
 - **“stream of consciousness” on IRC**
 - **Conservative**
 - **High quality design**



Retrospective

- ◆ Do different
 - Co-locate first sprint
 - Reduce turnover
 - ➔ Better support for employee development
 - ➔ Better support to prevent “burnout”
 - Secrecy
 - Language experimentation
 - Start earlier



Public release

Ubuntu Building Own Display Server, Unity To Switch to Qt/QML | OMG! Ubuntu! - Chromium

WebMail x Edit Profile | LinkedIn x 'New Unity Stack' App x OMG! Ubuntu! | Everyt x Canonical Targets Mobil x Ubuntu Building Own x

www.omgubuntu.co.uk/2013/03/canonical-announce-custom-display-server-mir-not-wayland-not-x

OMG! UBUNTU!

CATEGORIES ▾ UBUNTU TOUCH SUBMIT A TIP ▾ Search OMG! Ubuntu!

the new **UBUNTU ALL-IN-ONE PC** from hp

Ubuntu Building Own Display Server, Unity To Switch to Qt/QML

BY JOEY-ELIJAH SNEEDON UNDER DEV, NEWS, UBUNTU PHONE, UBUNTU TABLET MARCH 4, 2013

SHARE Like 426 Tweet 255 +1 243 258 points

Canonical has today publicly confirmed that they are working on a new cross-platform display server for Ubuntu.

Called 'Mir', the X Window Server replacement is tasked with 'enabling development of the next generation Unity'. Which, in yet another about-turn, is to be rebuilt in Qt/QML.

The news isn't much of a surprise. Earlier this year Canonical's Jono Bacon made several remarks in a Q&A session that hinted at the possibility of an alternative display manager.

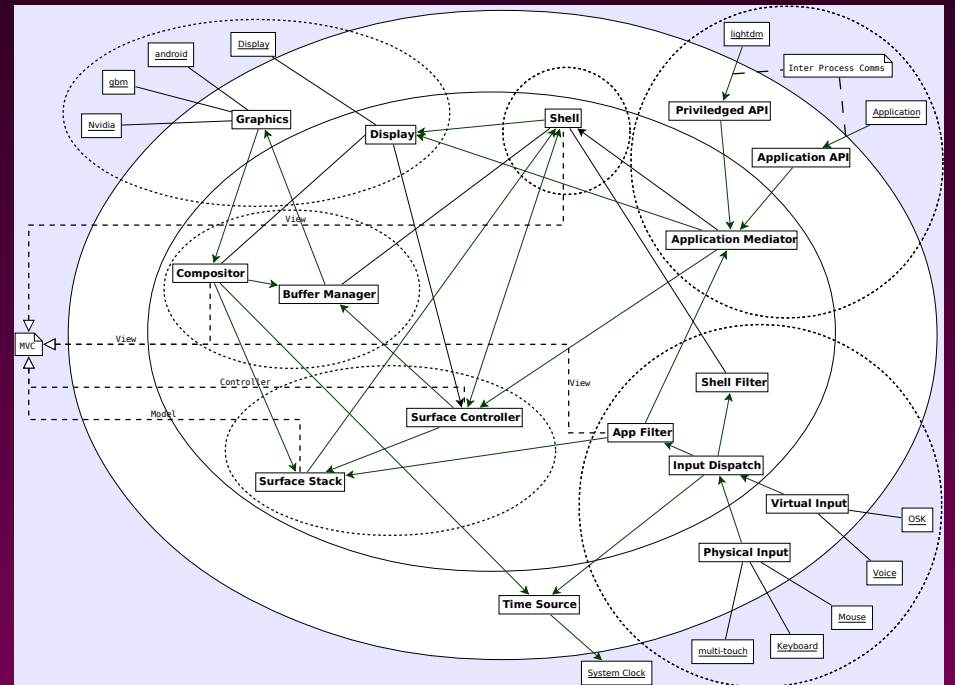
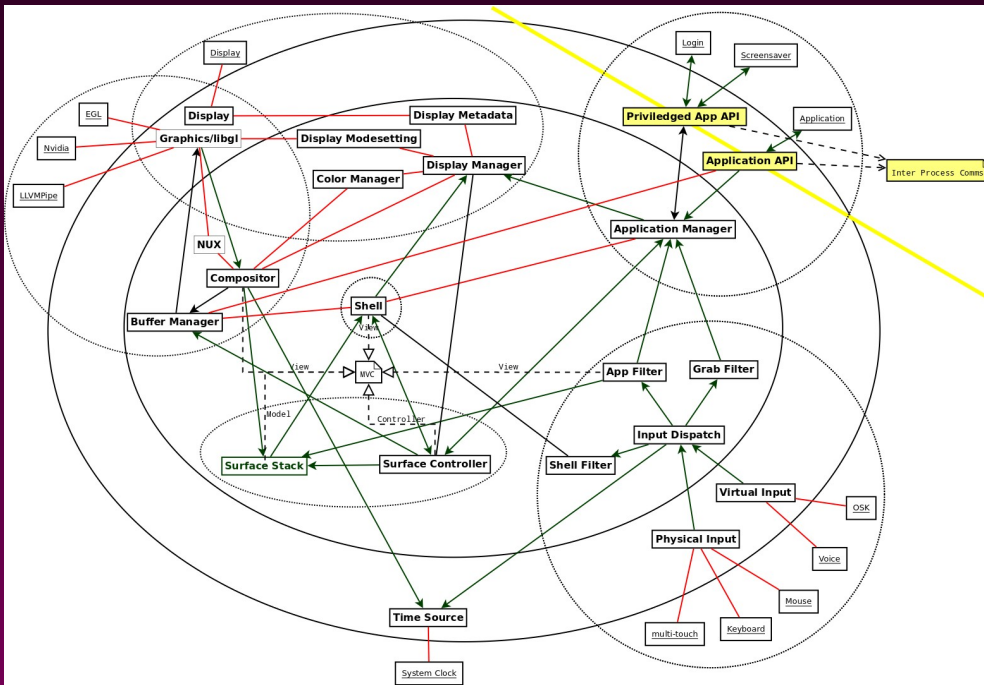
"...The simple reality is that X doesn't meet those needs, Wayland doesn't meet those needs."

From looking at the commit log for Mir this opinion has been held since June last year, which is when work on Mir appears to have begun.

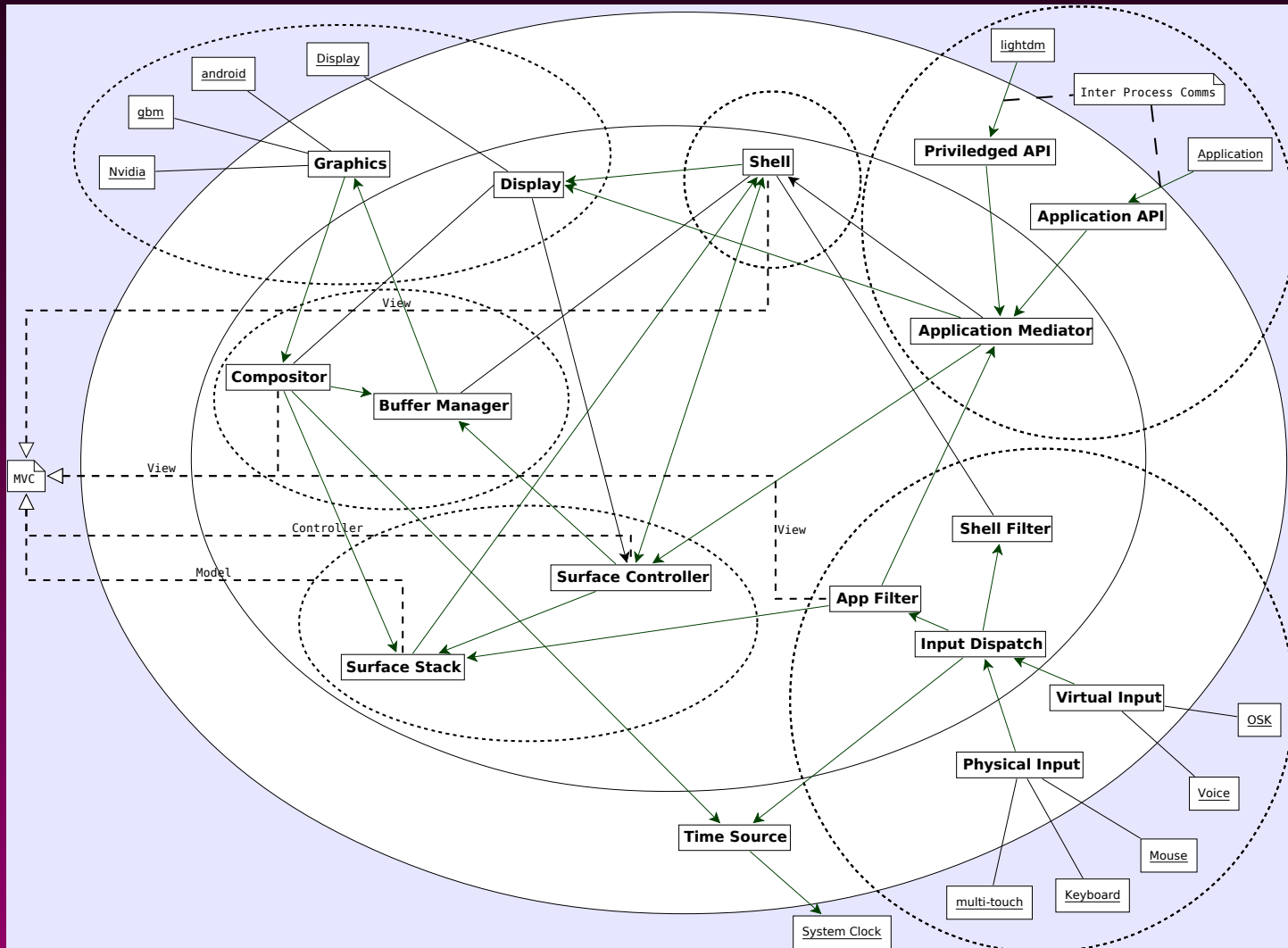
Popular posts

- It's Official: Ubuntu GNOME Remix Joins the Ubuntu Family
- Ubuntu Touch-Style Launcher Icons for the Ubuntu Desktop
- Ubuntu 13.04 Updates Nautilus, Update Tool Icons
- 'New Unity Stack' Approved for Ubuntu 13.04, Will Bring 'Smarter'
- Multi-Monitor Fixes for Unity 2D Heading to Precise

Design update



Design update



Feedback

Can I just say (from poking around the codebase) that it's fantastic to see Canonical producing lovely C++ code like this!

Particularly I like:

- * Use of modern C++ features like scoped locks, smart pointers and auto types
- * Good use of namespaces and matching directory paths
- * Using protobuf for wire formats
- * Good quantity of tests and use of google mock
- * Nice clean CMake scripts
- * Minimal debian packaging rules

I have no constructive input to add! I just wanted to highlight these things.

Cheers

Pete

Links and References

◆ mir

- <https://launchpad.net/mir>
- <http://unity.ubuntu.com/mir/>
- <https://wiki.ubuntu.com/MirSpec>

◆ GOOS

- <http://www.growing-object-oriented-software.com/>

◆ TeamViewer

- <http://www.teamviewer.com>

◆ Canonical

- <http://www.canonical.com/>

◆ g++

- <http://gcc.gnu.org/>

◆ cmake

- <http://www.cmake.org/>

◆ Jenkins

- <http://jenkins-ci.org/>

◆ Skype

- <http://www.skype.com/>

◆ Android

- <http://www.android.com/>

◆ Dialogue sheets

- <http://www.softwarestrategy.co.uk/dlgsheets/>

◆ Movie

- <http://www.youtube.com/watch?v=7grEFrTBzus>