BAB III
METODOLOGI PENELITIAN

3.1. Subjek dan Objek Penelitian

Subjek dalam penelitian ini adalah bagaimana penulis merancang bangun sebuah *wrapper library* untuk menjembatani antar sistem-sistem operasi *UNIX-like*, sedangkan objek penelitian ini yaitu kompatibilitas antar sistem-sistem operasi *UNIX-like*.

3.2. Alat dan Bahan Penelitian

Rancang bangun *wrapper library* ini memerlukan beberapa perangkat keras dan perangkat lunak. Berikut adalah spesifikasi perangkat keras dan perangkat lunak serta layanan yang akan digunakan pada penelitian ini.

A. Perangkat keras

1. iMac Retina 5K 27-inch 2019

   Perangkat ini digunakan untuk mengembangkan dan menguji Bridge.sh pada sistem operasi macOS.

Tabel 3.1: Spesifikasi iMac Retina 5K 27-inch 2019

Komponen	**Spesifikasi**
Prosesor	Intel Core i5 (6 *core*, 3,1 GHz)
Grafis	AMD Radeon Pro 575X (4 GB)
Memori	DDR4 8 GB
Penyimpanan	Fussion Drive 1 TB

2. HP 14-r110TU

   Perangkat ini digunakan untuk mengembangkan dan menguji Bridge.sh pada sistem operasi Linux.

Tabel 3.2: Spesifikasi HP 14-r110TU

Komponen	**Spesifikasi**
Prosesor	Intel Core i3-4030U
Grafis	Intel HD Graphics 4400 (HSW GT2)
Memori	DDR3 12 GB (*dual channel*, telah ditingkatkan)
Penyimpanan	SATA 500 GB

3. Galaxy J1 Ace

   Perangkat ini digunakan untuk mengembangkan dan menguji Bridge.sh pada sistem operasi Android.

Tabel 3.3: Spesifikasi Galaxy J1 Ace

Komponen	**Spesifikasi**
Prosesor	Spreadtrum SC8830 (4 *core*, 1,5 GHz)
Grafis	Mali-400 MP
Memori	1 GB
Penyimpanan	5 GB internal, 8 GB eksternal

B. Perangkat lunak

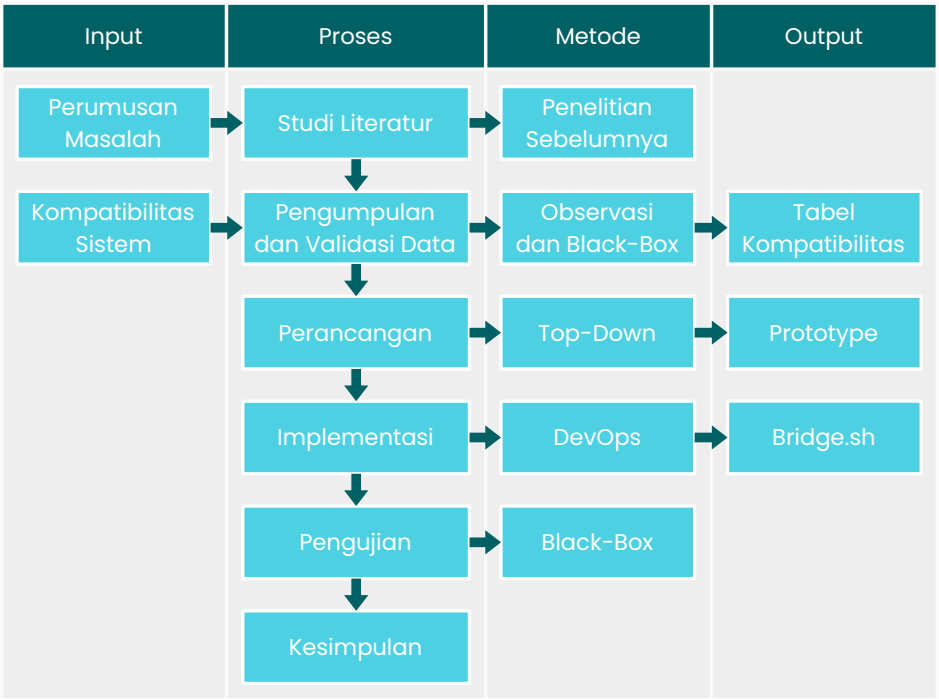
1. Sistem operasi
   a. Linux (berbagai distribusi)
   b. macOS
2. Alat pengembangan pada Linux (Android)
   a. Termux
3. Alat pengembangan pada Linux (desktop)
   a. GNOME Builder
   b. GNOME Terminal

4. Alat pengembangan pada macOS
    a. Visual Studio Code
    b. macOS Terminal
    c. VMware Fusion Player
5. Alat *continous integration* dan *continous delivery* (CI/CD)
    a. Git
    b. BATS

C. Layanan CI/CD
1. Github
2. Github Actions

3.3. Diagram Alir Penelitian

Tahapan penelitian dibuat secara sistematis untuk memudahkan penulis melakukan rancang bangun *library*. Berikut adalah diagram alir penelitian ini.



Gambar 3.1: Tahapan Penelitian dan Metode Penelitian

Pada Gambar 3.1 di atas, penulis menjelaskan tahapan penelitian yang akan dilakukan selama melaksanakan penitian untuk mendapatkan hasil penelitian yang terbaik. Berikut adalah penjabaran dari tahapan-tahapan yang penulis lakukan dalam penelitian ini.

3.3.1. Studi Literatur

Pada tahapan studi literatur, penulis mencari dan mengumpulkan literatur-literatur yang berkaitan dengan masalah-masalah yang ada dalam penelitian sebelumnya baik berupa artikel, buku, halaman web, dan/atau sumber-sumber lainnya.

3.3.2. Pengumpulan dan Validasi Data

Penulis menggunakan metode observasi untuk mengumpulkan data yang dibutuhkan untuk menemukan permasalahan para pengembang dalam pemrograman *shell* lintas *platform*. Setelah data dikumpulkan, penulis memvalidasi bagaimana perintah itu berjalan pada suatu sistem operasi. Untuk melakukannya, penulis menjalankan suatu perintah dengan suatu parameter dari suatu fitur, lalu mengobservasi hasil eksekusinya.

Hasil dari observasi dan validasi tersebut adalah tabel kompatibilitas dengan format sebagai berikut; di mana hijau menandakan bahwa fitur berjalan sesuai ekspektasi, kuning menandakan bahwa fitur secara parsial berjalan sesuai ekspektasi, dan merah menandakan bahwa fitur sama sekali tidak berjalan sesuai ekspektasi.

Tabel 3.4: Format Tabel Kompatibilitas

Perintah	**Fitur**	**Parameter**	**Linux**	**macOS**	**BSD**
Nama perintah, misal, **date**	Fitur yang perlu dijembatani, misal, parsing tanggal	Parameter yang digunakan, misal, **-d**	Kete-rangan	Kete-rangan	Kete-rangan
...	...		...	...	...
...	...		...	...	...

3.3.3. Perancangan

Untuk merancang *wrapper library* ini, penulis menggunakan metode *top-down design* dengan memperhatikan tabel kompatibilitas yang telah dibuat. Penulis memecah *wrapper library* menjadi beberapa komponen sebagai berikut:

A. *Installer*

Ini adalah sebuah skrip untuk mengunduh, memasang, serta mengonfigurasikan Bridge.sh.

B. *Startup*

Ini adalah sebuah skrip yang dijalankan saat log masuk *shell*[35], [36] atau saat *wrapper library* ini diimpor dalam suatu aplikasi – untuk memuat modul-modul yang diperlukan.

C. Modul inti

Modul ini berisi utilitas-utilitas internal, variabel-variabel internal, serta *shared code* untuk modul-modul lainnya.

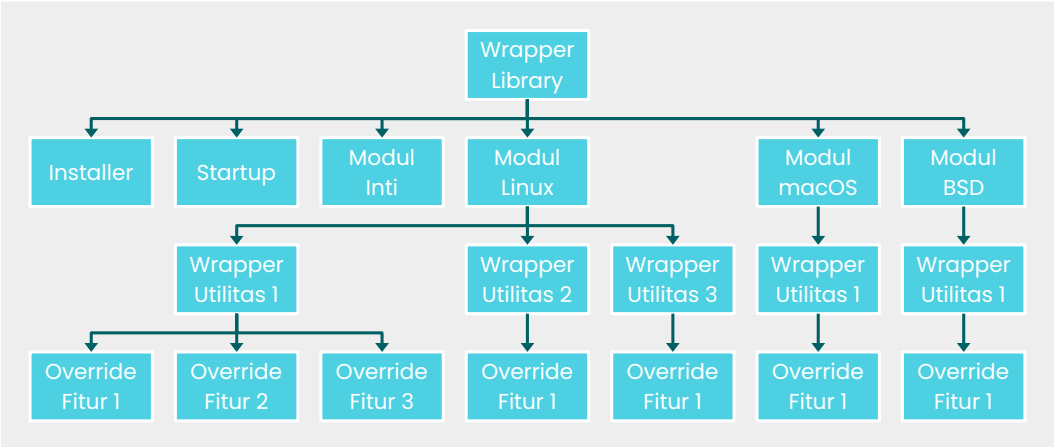
D. Modul kompatibilitas masing-masing sistem operasi

1. *Wrapper* masing-masing utilitas pada POSIX

   Wrapper ini membungkus utilitas yang asli pada sistem dengan antarmuka baru yang kompatibel dengan sistem *UNIX-like* lainnya.

   a. *Override* dari masing-masing fitur dari utilitas tersebut

      Override ini – menerjemahkan atau mengimplementasikan kembali – fitur yang ditimpa pada utilitas tersebut.



Gambar 3.2: Diagram *Top-Down* Perancangan Bridge.sh

3.3.4. Implementasi

Pada tahap ini, penulis akan mengimplementasikan *wrapper library* dari *scratch* menggunakan laptop pribadi penulis dan iMac di Lab Riset Institut Teknologi Telkom Purwokerto. Metode yang digunakan dalam implementasi *wrapper* ini adalah DevOps, karena Bridge.sh telah penulis jadikan sebagai proyek bersumber terbuka yang memungkinkan *continous workflow* dan dapat dikembangkan oleh banyak orang. Penulis menggunakan fasilitas CI/CD pada Github untuk proyek ini.

Penulis mulai melakukan implementasi setelah semua kebutuhan rancang bangun *wrapper library* terpenuhi. Untuk mempersiapkan CI/CD,

penulis membuat file-file konfigurasi yang diperlukan untuk CI/CD Github Actions. Setelah itu, untuk mengembangkan *wrapper library* ini, penulis mengimplementasikan setiap komponen pada diagram *top-down* perancangan Bridge.sh menggunakan bahasa *scripting* Bash dan Zsh.

Pada setiap modul kompatibilitas sistem operasi, penulis membuat lapisan kode (*shim*) yang menerjemahkan antarmuka utilitas *shell non-native* menjadi *native.* Lapisan kode tersebut terdiri dari *wrapper* untuk masing-masing utilitas. Di dalam *wrapper* tersebut, terdapat *override* untuk masing-masing fitur dari utilitas tersebut.

Selain komponen modul kompatibilitas sistem operasi, penulis juga mengimplementasikan komponen-komponen yang lain seperti modul inti, *startup*, serta *installer*. Komponen-komponen tersebut memiliki peran yang tidak kalah penting dengan komponen modul kompatibilitas. Komponen-komponen tersebut bersinergi satu sama lain agar *wrapper library* ini dapat digunakan dengan mudah oleh pengguna.

3.3.5. Pengujian

Pengujian *wrapper library* ini dilakukan menggunakan *black-box testing*, karena secara umum, pengujian di sini dilakukan menggunakan panggilan perintah dari luar sistem. Pengujian *black-box testing* pada penelitian ini meliputi *functional testing*, *non-functional testing*, dan *regression testing*. Pengujian tersebut dilakukan secara manual maupun otomatis menggunakan BATS untuk *unit testing*. Pengujian tersebut juga dilakukan secara *continous* dan simultan dengan implementasi dan *delivery* produk *wrapper library* ini.

Pengujian mulai dilakukan setelah implementasi mencapai tahapan di mana *wrapper library* telah dapat digunakan. Pada tahapan ini, pertama, penulis membuat kasus uji yang mencangkup *repetition*; *piping*; *IO redirection*; dan juga *error handling*. Penulis kemudian melakukan

pengujian secara manual dan kemudian mengimplementasikannya ke dalam *unit test*.

Setelah itu, pengujian akan berjalan secara otomatis setiap ada perubahan. Pada tahapan pengembangan (implementasi) tertentu, penulis akan melakukan pengujian *non-functional* secara manual. *Regression test* juga akan dilakukan pada tahapan tersebut untuk mengetahui dampak dari perubahan-perubahan yang telah dibuat.

3.3.6. Kesimpulan

Setelah melalui semua tahapan-tahapan sebelumnya, penulis mengambil kesimpulan untuk menjawab pertanyaan-pertanyaan penelitian yang telah penulis ajukan sebelumnya. Kesimpulan didapatkan dari analisis dan pengujian *wrapper library* yang dirancang dan dibangun. Setelah membuat kesimpulan, penulis memberikan saran untuk penelitian lebih lanjut dari *wrapper library* ini.