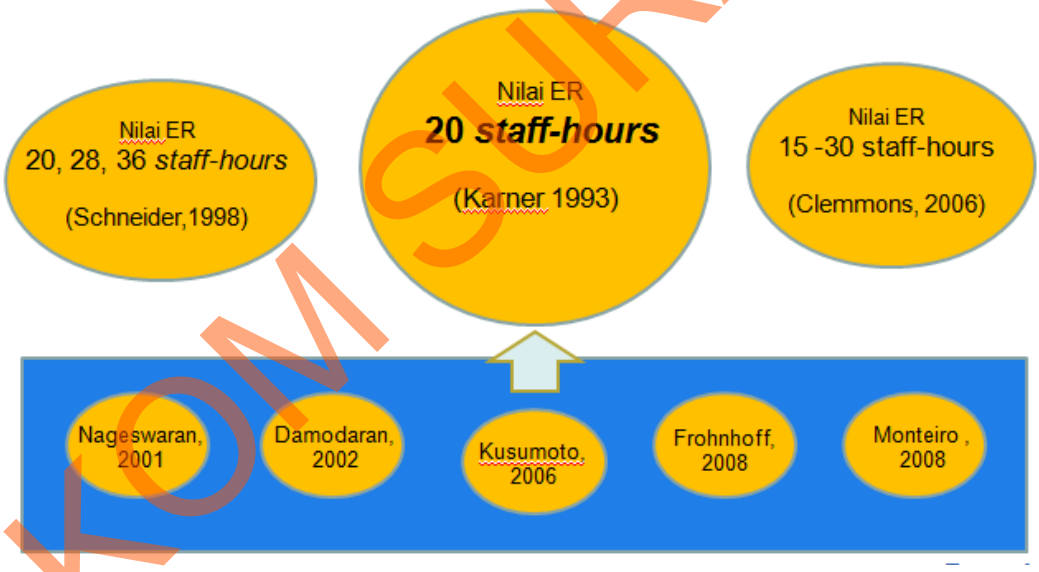
BAB II

LANDASAN TEORI

Pada bab ini dijelaskan mengenai teori-teori dan penelitian terkait terdahulu yang digunakan dalam pengerjaan tugas akhir ini.

2.1 Penelitian *Effort Rate* (ER) Sebelumnya

Pada penelitian sebelumnya telah dilakukan beberapa perhitungan untuk mencari nilai *effort rate*. Nilai *effort rate* yang dihasilkan dari penelitian sebelumnya menunjukkan hasil yang berbeda-beda. Penelitian tentang *effort rate* yang pernah dilakukan sebelumnya dapat dilihat pada gambar 2.1 seperti berikut:



Gambar 2.1 Penelitian *effort rate* sebelumnya

Dari gambar 2. 1 di atas, maka dapat disimpulkan bahwa nilai *Effort Rate* (ER) yang digunakan oleh sebagaian besar peneliti yaitu sebesar 20 *man-hours* seperti yang pertama kali diusulkan oleh Karner, namun penelitian tentang nilai *effort rate* yang dilakukan oleh Karner hanya menggunakan tiga data proyek pengembangan perangkat lunak dengan menggunakan analisis regresi. Analisis

regresi dengan menggunakan tiga data diskrit cenderung tidak akurat. Analisis korelasi antar data untuk membentuk persamaan regresi juga tidak dilakukan.

Selain itu, penelitian perhitungan *effort rate* yang dilakukan oleh Karner terjadi pada tahun 1993. Teknologi informasi dalam rentang waktu 1993 sampai 2013 mengalami pekembangan yang cukup pesat, sehingga sangat dimungkinkan nilai *effort rate* yang ditemukan oleh Karner tidak sesuai apabila diaplikasikan dalam perhitungan estimasi *effort* untuk proyek pengembangan perangkat lunak yang dikerjakan pada tahun 2013 dan tahun-tahun mendatang. Maka dari itu, nilai *Effort Rate* (ER) yang diusulkan oleh Karner dapat dipertanyakan dan ditinjau ulang.

2.2 Teori Pendukung

2.2.1 Estimasi *Effort*

Salah satu aspek terpenting dalam tahapan perencanaan adalah melakukan estimasi atau perkiraan, baik dari segi biaya, waktu maupun sumber daya. Definisi dari estimasi adalah sebuah pengukuran yang didasarkan pada hasil secara kuantitatif atau dapat diukur dengan angka tingkat akurasinya (Tockey, 2004).

Sisi penting estimasi dalam perencanaan proyek adalah munculnya jadwal serta anggaran yang tepat, meski tidak sepenuhnya sebuah estimasi akan berakhir dengan tepat. Tetapi, tanpa sebuah estimasi dalam pelaksanaan proyek perangkat lunak maka dapat dikatakan bahwa proyek perangkat lunak tersebut adalah sebuah *blind project*. Yang diibaratkan seperti seorang buta yang harus berjalan di sebuah jalan raya yang sangat ramai (Rizky, 2011).

Estimasi yang dilakukan pada penelitian ini diaplikasikan dalam proyek pengembangan perangkat lunak. Definisi dari estimasi perangkat lunak yaitu suatu

kegiatan melakukan prediksi atau ramalan mengenai keluaran dari sebuah proyek dengan meninjau jadwal, usaha, biaya bahkan hingga ke resiko yang akan ditanggung dalam proyek tersebut (Galorath, 2006). Meski estimasi tidak mungkin dapat menghasilkan sebuah hasil yang sangat akurat, tetapi ketidakakuratan tersebut dapat diminimalkan dengan menggunakan beberapa metode yang sesuai dengan proyek yang akan dilakukan estimasi.

Penelitian ini mengangkat estimasi *effort* pada proyek pengembangan perangkat lunak. *Effort* adalah kerja real yang kita lakukan dalam menyelesaikan suatu proyek. Satuannya adalah mandays atau manhour. Misalnya suatu aplikasi diestimasi membutuhkan effort 10 mandays. Artinya aplikasi ini akan selesai bila dikerjakan 1 orang selama 10 hari terus menerus atau 5 hari bila ada 2 pekerja. Effort tidak mempertimbangkan libur ataupun cuti (Muhardin, 2011).

Dari pengertian estimasi dan *effort* di atas, maka dapat disimpulkan bahwa estimasi *effort* adalah suatu kegiatan melakukan prediksi atau ramalan mengenai berapa banyak pekerja dan berapa lama waktu yang diperlukan untuk menyelesaikan proyek tersebut. Estimasi *effort* pada penelitian ini akan didapatkan setelah melakukan perhitungan menggunakan metode *use case point* (UCP).

2.2.2 *Use Case Point* (UCP)

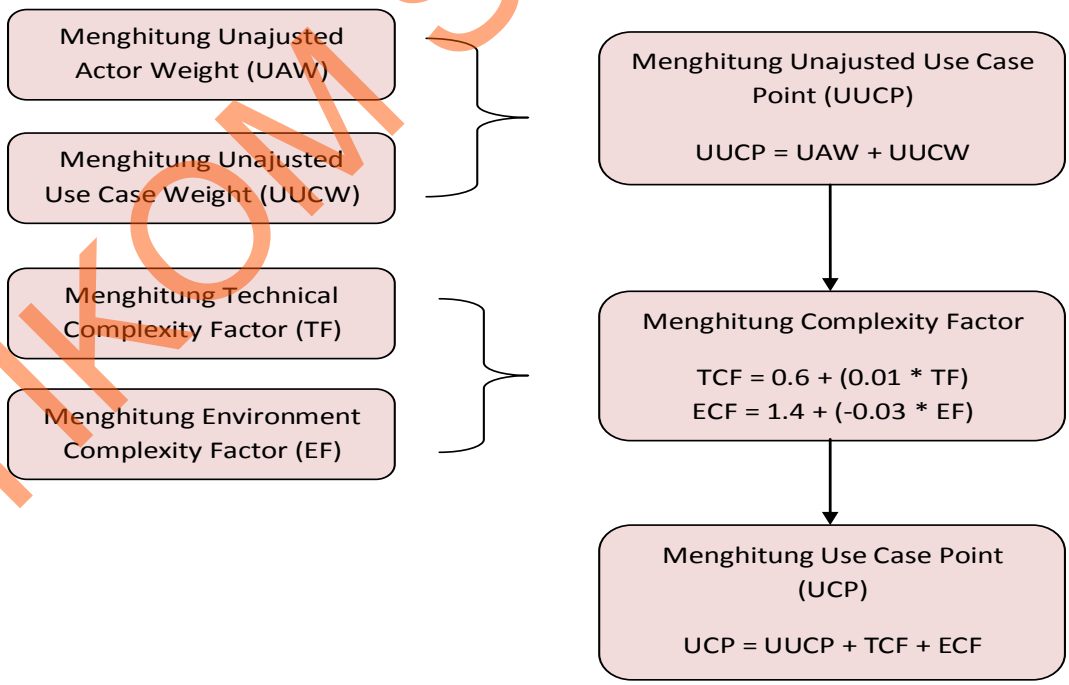
Metode *use case point* (UCP) adalah metode yang mempunyai kemampuan untuk memberikan estimasi *effort* yang diperlukan untuk membuat suatu proyek berdasarkan jumlah dan kompleksitas *usecase* yang dimiliki oleh proyek perangkat lunak tersebut (Karner, 1993). Menurut pendapat lain, UCP

adalah metode yang dapat menganalisa *actor, use case*, dan berbagai faktor teknis dan faktor lingkungan hingga menjadi suatu persamaan (Clemmons, 2006).

Kelebihan dari metode *use case point* yaitu dapat memberikan estimasi yang hampir mendekati estimasi sebenarnya yang dihasilkan dari pengalaman pembuatan atau pengembangan software. Hal tersebut dibuktikan oleh beberapa penelitian yang pernah dilakukan sebelumnya, dan menghasilkan pernyataan sebagai berikut:

1. UCP memiliki deviasi sebesar 6% (Nageswaran, 2001).
2. UCP memiliki deviasi sebesar 19%, sementara estimasi para ahli memiliki deviasi sebesar 20% (Anda, 2002).
3. UCP memiliki deviasi sebesar 9% (Carroll, 2005).

Langkah-langkah yang dilakukan dalam proses estimasi *effort* dengan *use case point* digambarkan dalam gambar 2.2 berikut ini (Karner, 1993) :



Gambar 2.2 Langkah – langkah Metode *Use Case Point* (UCP)

2.2.2.1 Menghitung *Unadjusted Use Case Point* (UUCP)

a). *Unadjusted Actor Weights* (UAW)

Langkah pertama adalah menentukan terlebih dahulu aktor sebagai *simple*, *average*, atau *complex* sesuai tabel 2.1 seperti berikut:

Tabel 2.1 Tipe, Bobot, dan Deskripsi *Actor*

Actor	Weight	Description
Simple	1	Didefinisikaan dengan API
Medium	2	Berinteraksi melalui Protokol TCP/IP
Complex	3	Berinteraksi dengan GUI atau Web Page

Total *Unadjusted Actor Weights* (UAW) didapat dari menghitung jumlah *actor* dari masing-masing jenis (tingkat kompleksitas), dikali dengan total faktor berat masing-masing sesuai dengan tabel.

b). *Unadjusted Use Case Weights* (UUCW)

Cara menghitung UUCW sama dengan cara menghitung UAW, yaitu masing-masing *use case* dibagi menjadi 3 kelompok yaitu *simple*, *average*, dan *complex*, tergantung dari jumlah transaksi yang dilakukan. Untuk penjelasan lebih detil tentang deskripsi *use case* dapat dilihat pada tabel 2.2 seperti berikut :

Tabel 2.2 Tipe, Bobot, dan Deskripsi *Use Case*

Use Case	Weight	Description
Simple	5	Menggunakan <= 3 transaksi
Medium	10	Menggunakan 4 sampai 7 transaksi
Complex	15	Menggunakan > 7 transaksi

Total *Unadjusted Use Case Weights* (UUCW) didapat dari menghitung jumlah *use case* dari masing-masing tingkat kompleksitas dikali dengan total faktor setiap *use case*. Kemudian jumlahkan UAW dan

UUCW untuk mendapatkan Unadjusted *Use Case Point* (UUCP), seperti rumus berikut :

$$UUCP = UAW + UUCW \text{..... (2.1)}$$

2.2.2.2 Menghitung *Technical Complexity Factor* (TCF) dan *Enviromental Complexity Factor* (ECF)

Pada perhitungan nilai *Use Case Point* (UCP) terdapat nilai *complexity factor*. Pengertian dari *complexity factor* adalah faktor-faktor yang berpengaruh secara langsung dalam proses pengerjaan proyek perangkat lunak tersebut. *Complexity factor* dibagi menjadi 2 kelompok, yaitu :

1. *Technical Complexity Factor* (TCF)
2. *Environmental Complexity Factor* (ECF)

Berikut penjelasan masing-masing dari *complexity factor* :

a). *Technical Complexity Factor* (TCF)

Tabel 2.3 *Technical Factor* dan Bobot

Technical Factor		Bobot
1.	Distributed System Required	2
2.	Response Time is Important	1
3.	End User Efficiency	1
4.	Complex Internal Processing Required	1
5.	Reusable Code Must Be A Focus	1
6.	Installation easy	0.5
7.	Usability	0.5
8.	Cross-platform support	2
9.	Easy to change	1
10.	Highly concurrent	1
11.	Custom security	1
12.	Dependence on third-part code	1
13.	User training	1

Nilai-nilai pada *technical factor* tersebut dikalikan dengan bobot nilai masing-masing. Bobot nilai yang diberikan pada setiap faktor tergantung dari seberapa besar pengaruh dari faktor tersebut. 0 berarti tidak mempengaruhi, 3 berarti rata-rata, dan 5 berarti memberikan pengaruh yang besar. Hasil perkalian nilai dan bobot tersebut kemudian dijumlahkan untuk mendapatkan total *Technical Factor* (TF), yang kemudian digunakan untuk mendapatkan *Technical Complexity Factor* (TCF).

$$TCF = 0.6 + (0.01 \text{ x } TF) \text{ ..... } (2.2)$$

b). *Enviromental Complexity Factor* (ECF)

Tabel 2.4 *Environmental Factor* dan Bobot

	Enviromental Factor	**Bobot**
1.	Familiarity with the Project	1.5
2.	Application Experience	0.5
3.	OO Programming Experience	1
4.	Lead Analyst Capability	0.5
5.	Motivation	1
6.	Stable Requirements	2
7.	Part Time Staff	-1
8.	Difficult Programming Language	-1

Nilai-nilai pada *environmental factor* tersebut dikalikan dengan bobot nilai masing-masing. Bobot nilai yang diberikan pada setiap faktor tergantung dari seberapa besar pengaruh dari faktor tersebut. 0 berarti tidak mempengaruhi, 3 berarti rata-rata, dan 5 berarti memberikan pengaruh yang besar. Hasil perkalian nilai dan bobot tersebut kemudian dijumlahkan untuk mendapatkan total *Environmental Factor* (EF), yang

kemudian digunakan untuk mendapatkan *Environmental Complexity Factor* (ECF).

$$\text{ECF} = 1.4 + (-0.03 \text{ x EF}) \text{ ..... (2.3)}$$

Sehingga akhirnya kita bisa mendapatkan nilai dari *Use case Point* (UCP) yang didapatkan melalui perkalian UUCP, TCF, dan ECF.

$$\text{UCP} = \text{UUCP} + \text{TCF} + \text{ECF} \text{..... (2.4)}$$

2.2.3 Perhitungan Nilai *Effort Rate*

Effort rate didefinisikan sebagai jumlah usaha per *use case point*. Pendekatan yang dijelaskan bersifat umum dan dapat digunakan untuk menganalisa berbagai data, tidak hanya data untuk pengembangan perangkat lunak, tetapi juga data pemeliharaan perangkat lunak dan jenis lain dari rekayasa perangkat lunak (Stewart, 2002).

Effort rate adalah rasio jumlah jam orang per *use case point* berdasarkan proyek-proyek di masa lalu. Jika proyek tersebut merupakan proyek baru dan tidak terdapat data histori yang telah terkumpul, maka digunakan nilai yang berkisar antara 15 sampai 30. Namun, nilai yang paling sering dipakai adalah angka 20 (Clemmons, 2006).

Rumus perhitungan estimasi *effort* menggunakan metode UCP adalah sebagai berikut :

$$\text{Estimasi Effort} = \text{UCP x ER} \text{ ..... (2.5)}$$

Apabila nilai ER dihitung dari satu proyek saja maka nilai ER didapatkan dari pembagian antara nilai *actual effort* dengan nilai UCP, sebagai berikut :

$$Effort\ Rate = \frac{Actual\ \ Effort}{\text{UCP}} \text{ ..... (2.6)}$$

Namun, pada penelitian tugas akhir ini dilakukan perhitungan nilai *effort rate* menggunakan beberapa data proyek pengembangan perangkat lunak, sehingga untuk mendapatkan nilai ER yang valid harus dilakukan perhitungan menggunakan persamaan regresi.

2.2.4 Analisis Korelasi dan Persamaan Regresi

2.2.4.1 Analisis Korelasi

Analisis korelasi merupakan analisis terhadap kekuatan hubungan antara variabel bebas X dengan variabel tak bebas Y. Koefisien korelasi linier adalah ukuran hubungan linier antara satu variabel x dengan satu variabel y, dan dilambangkan dengan "r" (Usman, 2006). Hasil dari perhitungan korelasi diinterpretasikan pada sebuah hubungan yang didasarkan pada nilai angka yang muncul. Interpretasi nilai korelasi dapat dilihat pada tabel 2.5 seperti berikut :

Tabel 2.5 Interpretasi Nilai R (Korelasi)

R	Interpretasi
0	Tidak Berkorelasi
>0 – 0.25	Korelasi Sangat Lemah
>0.25 – 0.5	Korelasi Cukup
>0.5 – 0.75	Korelasi Kuat
>0.75 – 0.99	Korelasi Sangat Kuat
1	Korelasi Sempurna

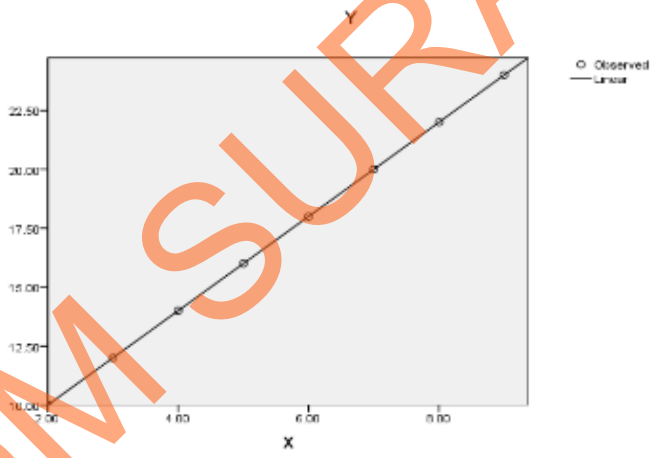
2.2.4.2 Persamaan Regresi

Analisis regresi merupakan hubungan ketergantungan antara satu variabel tak bebas (*dependent* variabel) dengan satu atau lebih variabel bebas (*independent* variabel) dengan tujuan untuk memperkirakan nilai rata-rata dari variabel tak

bebas, apabila variabel bebasnya sudah diketahui (Usman, 2006). Variabel bebas dilambangkan dengan X dan variabel tak bebas dilambangkan Y. Berikut persamaan matematiknya:

$$y = a + bx \quad \text{..... (2.7)}$$

Untuk mengetahui hubungan antara variabel bebas dengan variabel tak bebas dimulai dengan mencari bentuk terdekat dari hubungan tersebut dalam sebuah diagram pencar, dimana setiap datanya dinyatakan dalam bentuk koordinat (x,y). Jika titik-titik yang terbentuk mengikuti suatu garis lurus, maka variabel x dan y dikatakan saling berhubungan secara linier (Usman, 2006), seperti gambar 2.3 berikut ini:



Gambar 2.3 Pola Garis Lurus Regresi

Antara variabel bebas X dan variabel terikat Y membentuk sebuah pola garis yang lurus, dan dalam aplikasinya jika nilai X meningkat maka nilai Y juga meningkat dan jika nilai X mengalami penurunan makan nilai Y juga mengalami penurunan.

2.2.5 Proyek Perangkat Lunak Pemerintahan

Pemanafaatan internet dalam suatu institusi dapat membuat pekerjaan semakin efektif. Untuk dinas pemerintahan, internet akan sangat membantu dalam menyukseskan program *e- government*. Dalam *e-government*, internet menjadi

teknologi yang berperan dalam proses penyediaan dan transfer informasi dari pemerintah kepada pihak lain, misalnya warga masyarakat, ataupun sebaliknya.

Pemanfaatan teknologi komunikasi dan informasi dalam proses pemerintahan akan meningkatkan efisiensi, efektifitas, transparansi dan akuntabilitas penyelenggaraan pemerintahan. *E-government* merupakan perubahan radikal di dalam sistem dan tata laksana pemerintahan yang menuntut teladan kepemimpinan, kesediaan merubah paradigma, berani bertindak transparan, dan semua itu bukan sekedar untuk melayani kepen-tingan publik semata, tetapi mencakup kepentingan yang lebih luas yaitu sebagai bagian dari sistem pemerintahan yang bertujuan untuk mensejahterakan masyarakat (Wigrantoro, 2003).

Pada penelitian ini, studi kasus yang diteliti yaitu studi kasus pembuatan perangkat lunak di bidang kepemerintahan, antara lain :

1. Pembuatan perangkat lunak website Pemerintah Kabupaten Buton Utara (www.butonutarakab.go.id).
2. Pembuatan perangkat lunak Bursa Kerja online Dinas Tenaga Kerja Pemerintah Kota Surabaya (bursakerja.surabaya.go.id).
3. Pembuatan perangkat lunak website Badan Kependudukan dan Keluarga Berencana Yogyakarta (yogya.bkkbn.go.id).
4. Pembuatan perangkat lunak website resmi Pemerintah Kabupaten Tegal (www.tegalkab.go.id).
5. Pembuatan perangkat lunak website Dinas Pertanian D.I.Yogyakarta (www.distan.pemda-diy.go.id).

6. Pembuatan perangkat lunak website Dinas perindustrian, perdagangan, dan koperasi DIY (www.disperindagkop.pemda-diy.go.id).
7. Pembuatan perangkat lunak website Gerai Pelayanan Perizinan Terpadu BKPM Provinsi DIY (www.geraip2t.jogjaprov.go.id).
8. Pembuatan perangkat lunak website Dinas Kesehatan Kabupaten Tegal (www.dinkes.tegalkab.go.id).