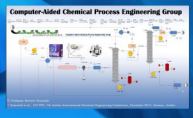
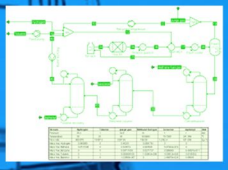
MODELING dan SIMULASI

# PADA PROSES INDUSTRI KIMIA

oleh
SETIA BUDI SASONGKO





UNDIP PRESS
SEMARANG

# Modeling dan Simulasi pada Proses Industri Kimia

**Oleh :**

**Setia Budi Sasongko**

UNDIP PRESS
SEMARANG
2020

# Modeling dan Simulasi pada Proses Industri Kimia

**oleh :**
**Setia Budi Sasongko**

**x + 138 hlm (Uk. 15.5cm x 23cm)**
**ISBN : 978-979-0977-54-9**

Desember 2020

diterbitkan oleh :
UNDIP PRESS
SEMARANG

# Kata Pengantar

Pertama-tama, penulis mengucapkan puji syukur kehadlirat Allah SWT, yang telah memberi semua kebaikan, sehingga buku ini dapat diselesaikan. Selain itu, penulis juga mengucapkan terima kasih kepada semua yang telah membantu dalam penyelesaian buku ini baik moral maupun material.

Buku ini merupakan hasil penelitian yang bersifat *dry-lab* artinya penelitian pengembangan perangkat lunak khususnya untuk perangkat lunak yang bersifat bebas bayar. Penelitian dalam bidang perangkat lunak sudah menjadi bagian yang tidak dapat ditinggalkan untuk saat ini di era digital atau era hampir disemua bidang menggunakan alat bantu komputer dalam berbagai macam bentuk, baik telepon genggam, alat-alat pengendali jarak jauh, maupun komputer itu sendiri. Permasalahannya masih banyak buku dalam bidang perangkat lunak yang masih membahas bidang ilmu secara umum. Oleh karenanya, pembahasan pada buku ini lebih ditekankan pada bidang Teknik Kimia. Meskipun perangkat lunak simulasi Teknik Kimia sudah berkembang cukup pesat, akan tetapi hampir sebagian besar perangkat lunak tersebut berbayar. Selain itu perangkat lunak simulasi bersifat black-box, artinya filosofi keilmuan khususnya bagi mahasiswa kurang, sehingga penggunaannya bersifat *trial-error*. Penulisan program komputer didekati dari dua sisi, yaitu sisi logika penyelesaian program komputer dan sisi validasi dalam bidang keilmuan Teknik Kimia.

Buku ini ditulis dengan menyusun dan mengeksekusi program terlebih dahulu, yang dilakukan dengan berbagai macam eksperimen, sehingga didapat hasil yang memuaskan. Setelah itu, baru ditulis dalam bentuk narasi dan logika penulisan ilmiah, sehingga hasil dari tulisan ini dapat diulang dan dapat

dikembangkan untuk kemajuan penelitian perangkat lunak khususnya bidang Teknik Kimia.

Akhir kata, semoga buku ini dapat menjadi rujukan dan pengembangan digitalisasi Teknik Kimia. Tiada gading yang tak retak, kritik dan saran perbaikan penulis harapkan untuk perbaikan dan pengembangan selanjutnya.

Setia Budi Sasongko
sbudisas@live.undip.ac.id

# Daftar Isi

# Daftar Tabel

# Daftar Gambar

# BAB I

# Matematika modeling dan simulasi numerik sistem blok

Setia Budi Sasongko (sbudisas@live.undip.ac.id)

Kata kunci (*Keywords*):
Scilab, *Free Open Source Software* (FOSS), matematika modeling dan simulasi.

Abstraksi

Pada artikel ini, telah dituliskan program komputer dengan menggunakan Scilab 5.5.2, yang merupakan program komputer yang bersifat gratis dengan menggunakan sistem blok artinya program yang dibuat dalam bentuk simple, interkoneksi dan flesibel. Kelebihan dari program ini, algoritmanya mudah difahami, dan apabila ada kesalahan akan mudah untuk dilacak.

Artikel ini merupakan hasil penelitian yang bersifat dry-lab, oleh karenanya format penulisan lebih pada artikel penelitian. Tujuan utama dari artikel ini adalah mengembangkan program Scilab dalam bentuk blok-blok, sedangkan sistem merupakan kumpulan blok-blok yang saling berinteraksi.

Diawali dengan penurunan model matematis, penyelesaian model dengan metoda numerik dalam bentuk sistem blok, penentuan parameter model dengan metod pencocokan data experimen (khusus untuk sistem bioproses), serta simulasi model dengan mengubah-ubah variabel telah dilakukan dengan hasil sangat memuaskan. Beberapa contoh kasus telah dipresentasikan pada artikel ini, dilengkapi dengan source-code program Scilab sehingga dapat dikembangkan atau digunakan oleh peneliti lain.

## 1. Pendahuluan.

Dalam pemodelan matematika pada dasarnya adalah penggabungan dari persamaan-persamaan dasar, menjadi suatu sistem sesuai dengan tujuan, kemudian diselesaikan dengan cara numerik sehingga dapat digunakan untuk memprediksi atau mensimulasi sehingga karakteristik dari sistem proses dapat diketahui. Pemodelan dan simulasi ini disebut dengan *dry-lab* karena laboratorium yang dimaksud disini adalah komputer beserta perangkat lunaknya.

### 1.1. Permasalahan dan tujuan

Perangkat lunak siap pakai (CAD – *Computer Aided Design*) semakin berkembang dengan pesat. Kelebihan dari perangkat lunak siap pakai (berbayar) cukup banyak, antara lain mempunyai data base yang cukup lengkap, sangat mudah digunakan (*user friendly*), akan tetapi ada juga kekurangan antara lain harga yang relative mahal, dan pengguna tidak atau kurang memahami filosofi (latar belakang) dari sistem proses yang digunakan dalam bentuk persamaan matematik.

Untuk mengatasi hal tersebut, maka perlu dikembangkan program-program sistem proses dalam bentuk persamaan matematika. Model persamaan matematik pada dasarnya merupakan hubungan antar variabel dalam sistem sehingga dapat menggambarkan karakteristik dari sistem proses. Permasalahan selanjutnya adalah kesulitan dalam penyelesaian model matematis secara analitis. Untuk mengatasi hal tersebut, digunakan penyelesaian secara numeris (pendekatan). Persamaan matematika akan diubah dalam dalam bentuk aritmatika (tambah, kurang, kali dan bagi) atau bentuk aljabar yang sudah dibuat baku. Bentuk tersebut pada umumnya memerlukan perulangan sehingga diperlukan alat bantu komputer untuk mendapatkan

hasil mendekati nilai analitis.

Permasalahan pada pemrograman, pertama pada umumnya kode program sering tidak ditampilkan, kedua program diselesaikan tidak terstruktur, artinya kode program menjadi satu bagian. Oleh karenanya pada tulisan ini akan disusun program-program dalam bentuk blok-blok kecil, untuk program kompleks dapat digabungkan berdasrkan pada blok-blok tersebut. Keuntungan dari sistem blok ini pertama untuk pembaca kode program akan lebih mudah memahami algoritma program, kedua akan lebih mudah dalam mendeteksi kesalahan.

Pada sistem model sering diketemukan bentuk persamaan non linear, sebagai contoh pada sistem bioproses, kinetika pertumbuhan dari mikroba. Sehingga untuk mendapatkan parameter estimasi dari hasil percobaan perlu dilakukan linearisasi persamaan. Pada tulisan ini akan dibahas program untuk mendapatkan parameter berdasarkan linear dan non linear persamaan.

Tujuan utama pada bagian ini akan dikembangkan pemodelan matematika secara umum, kemudian akan diselesaikan secara numerik dengan menggunakan perangkat lunak **Scilab 5.2.2** yang bersifat ***free*** dan ***open source software*** **(FOSS)** dan program yang dikembangkan bersifat **blok**. Penggunaan FOSS dapat dikembangkan lebih luas oleh segala lapisan peneliti secara legal. Pembuatan program secara modular, maka untuk mengembangkan sistem yang lebih besar, akan menjadi lebih mudah. Tulisan ini menekankan pada penyusunan program komputer yang bersifat modular atau blok, sehingga dapat dikembangkan untuk sistem yang lebih kompleks dan luas sebagai gabungan dari blok-blok yang tersedia. Tulisan ini diperuntukan bagi mahasiswa, peneliti pengembang perangkat lunak.

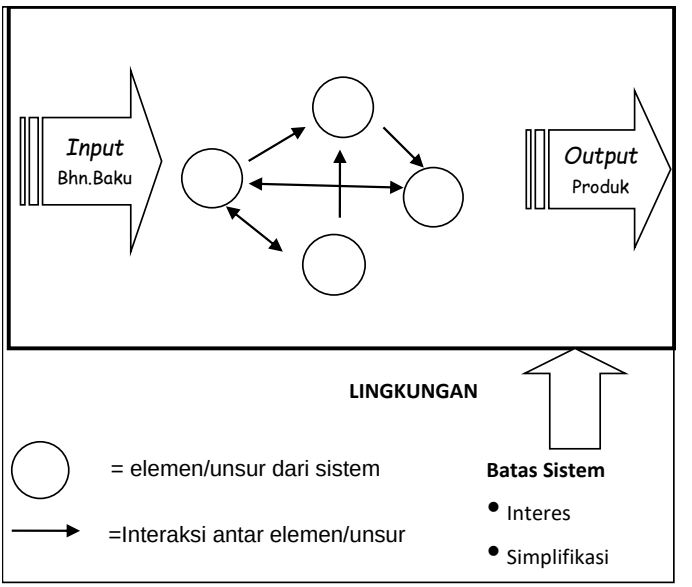
## 2. Metoda

Model matematis dikembangkan dari berbagai referensi, demikian juga dengan data diambil dari beberapa jurnal. Penyelesaian model matematis dalam bentuk program komputer diselesaikan secara metoda numeris dengan menggunakan perangkat lunak Scilab 5.5.2 versi Window 64 bits (Sasongko, 2010). Penggunaan Scilab versi 5.5.2 lebih stabil dibanding Scilab versi 6.02 (phoudha, 2020). Scilab versi 5.5.2 dapat diunduh pada alamat: https://www.scilab.org/download/5.5.2/scilab-5.5.2_x64.exe. Setelah model jadi, tahap selanjutnya adalah simulasi.

Simulasi merupakan aktivitas menirukan suatu kejadian, keadaan atau sistem yang sesungguhnya dengan demikian pengguna akan mendapatkan gambaran atau fenomena sistem yang dikaji. Dalam hal ini sistem yang disimulasikan adalah sistem proses industri kimia. Untuk melaksanakan simulasi ini diperlukan simulator, yang bentuknya dapat berbagai macam, akan tetapi pada artikel ini akan dibahas simulator dengan bantuan (berbasis) komputer, sebagai alat bantu untuk mengkaji sistem proses yang dikaji. Proses sebagai bagian dari sistem, yang akan merubah masukan dari proses berupa bahan baku (***input*****)** menjadi keluaran (***output***) berupa produk. Kata merubah atau tranformasi dalam proses dapat mempunyai bermacam-macam arti, dapat merubah dalam artian fisis, kimiawi, biologis atau bahkan lebih luas lagi seperti dari segi ekonomi. Hubungan dari ketiganya, dapat dituliskan sebagai berikut: *input* → **proses** → *output*.

Misalnya pada pabrik (industri) kimia, peralatan-peralatan pada pabrik dapat terdiri dari tangki, pompa, reaktor, alat-alat pemisah dan lain sebagainya sebagai elemen dari sistem tersebut. Sehingga kesemuanya bagian tersebut dapat disebut dengan sistem proses. Disisi lain, reaktor ataupun pompa sendiri dapat

juga merupakan sistem proses juga. Oleh karena itu dalam membahas sistem dari proses perlu ada pembatas /batasan (*boundary*), dimana diluar batasan sistem tersebut disebut dengan lingkungan (*environment*). Hal ini dapat dilihat pada gambar 1.1 merupakan kesatuan dari sistem proses yang terdiri dari elemen-elemen proses, masukan (*input*), keluaran - produk (*output*), interaksi antar elemen, garis batas sistem serta lingkungan. Jadi sistem proses dapat berupa sistem yang besar atau sistem yang kecil tergantung dar objek yang akan dikaji.




Gambar 1.1: Sistem Proses sebagai suatu kesatuan.

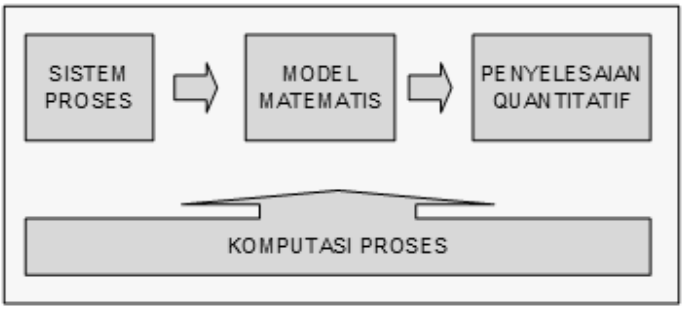
Simulator pada dasarnya merupakan suatu model. Dimana model dapat juga dikatakan sebagai tiruan / imitasi yang menggambarkan atau mendeskripsikan dari sistem proses, dalam hal ini dapat berupa tiruan dalam bentuk yang lebih kecil atau bentuk yang lebih sederhana, agar dapat dipelajari karakteristiknya dan pada umumnya dilakukan dalam

laboratorium. Dalam hal ini merupakan model tersebut dapat disebut juga model-fisik. Karena ukurannya relatif lebih kecil, sehingga disebut dengan skala laboratotium atau kadang-kadang untuk ukuran yang relatif lebih besar disebut dengan *pilot plant*. Salah satu kendala dari model-fisis, biayanya relatif mahal.

Selain model yang telah dibahas tersebut, ada juga model bentuk lain berupa model-matematis. Model matematis ini mengimitasikan (menggambarkan) sistem proses untuk mengetahui karakteristiknya. Selain itu, model matematis tersebut dapat juga digunakan sebagai perancangan dari sistem proses. Model matematis pada dasarnya merupakan sekumpulan persamaan matematis, variabel maupun konstanta. Oleh karenanya model matematis berfungsi sebagai:

1. Sintesa sistem proses atau satuan operasi.
2. Analisis sistem proses dan optimisasi
3. Pemilihan atau pengembangan sistem proses

Akan tetapi model matematis tidak dapat berfungsi sebagaimana yang ditunjukkan diatas apabila belum dilengkapi dengan penyelesaian dari model matematis tersebut, atau dapat dikatakan sebagai penyelesaian dari kumpulan persamaan yang ada, sehingga akan didapat hasil yang bersifat kuantitatif. Kesemua bagian tersebut dikelompokkan dalam pembahasan dari komputasi proses, sebagaimana yang ditunjukkan pada gambar 1.2. Hasil penyelesaian berupa variabel-variabel terukur secara kuantitatif dibandingkan dengan variabel terukur secara riil di lapangan. Apabila perbedaan antara keduanya tidak terlalu jauh, maka model dapat dikatakan mendekati keadaan yang sebenarnya.




Gambar 1.2: Ruang lingkup bahasan dari komputasi proses

Berdasarkan gambar 1.2 tersebut diatas, tahap pertama adalah melihat permasalahan yang ada pada sistem proses, diikuti dengan penentuan tujuan dari sistem proses yang akan dianalisis. Tahap berikutnya yang cukup penting adalah pemodelan matematis yaitu menyusun persamaan-persamaan. Dimana persamaan tersebut didapat berdasarkan kumpulan dari teori (konsep dasar) yang masih relevan, pengalaman baik di lapangan (pabrik) maupun dari hasil penelitian yang disusun berdasarkan hubungan sebab-akibat. Tahap selanjutnya adalah penyelesaian dari persamaan-persamaan matematis yang umumnya menggunakan metoda pendekatan atau lebih sering disebut dengan metoda numerik. Penyelesaian secara analitis mempunyai tingkat kesulitan yang cukup tinggi, apalagi apabila jumlah maupun bentuk persamaannya cukup banyak. Pada umumnya penyelesaian dengan metoda numerik dapat dibagi menjadi tiga bagian besar, yaitu sekuensial (berurutan), kondisional dan perulangan. Untuk mendapatkan hasil perhitungan dengan metoda numerik mendekati atau sama dengan hasil dengan perhitungan secara analitis, maka diperlukan proses perulangan. Oleh karenanya, diperlukan alat bantu hitung berupa komputer. Dengan demikian dimulai dari penyusunan model matematis sampai dengan penyelesaian model matematis tersebut

diperlukan kejelian, pengalaman untuk menyelesaikan model matematis dengan metoda numerik berupa penyusunan urutan yang logis dalam bentuk algoritma sehingga akan didapat hasil yang stabil. Yang dimaksud dengan hasil yang stabil adalah apabila proses perhitungan dieksekusi dilain waktu, maka akan didapat hasil yang sama.

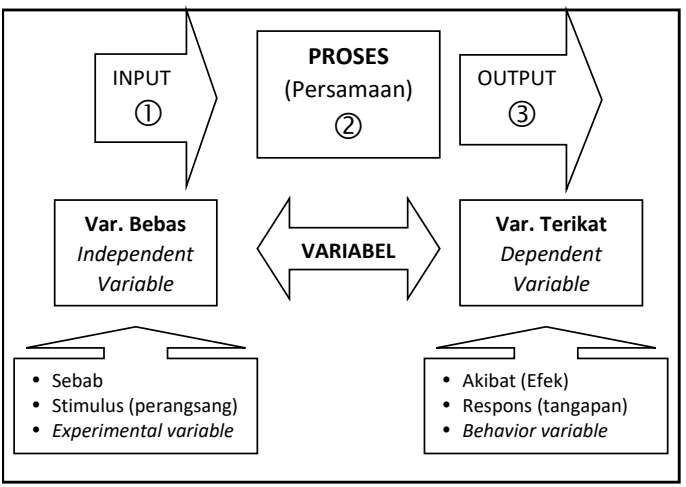
Model matematis sebagai kumpulan dari persamaan-persamaan, variabel maupun konstanta yang berfungsi untuk menggambarkan fenomena dari sistem proses. Berikut beberapa elemen dalam model matematis beserta definisinya antara lain:

- **Variabel** adalah bagian dari model matematis dimana nilainya dapat berubah-ubah (tidak tetap).
- **Parameter** adalah bagian dari model matematis yang nilainya relatif tetap pada kondisi tertentu.

Misalkan konstanta gravitasi mempunyai nilai tetap pada tempat tertentu akan tetapi nilainya dapat berubah ditempat lain. Contoh lain konstanta kecepatan reaksi berdasarkan persamaan Arrhenius: $k = A.\exp^{-E/RT}$, nilai *k* tersebut akan tetap apabila pada kondisi isotermal (suhu tetap), akan tetapi apabila suhunya berubah, maka nilai *k* akan berubah juga.

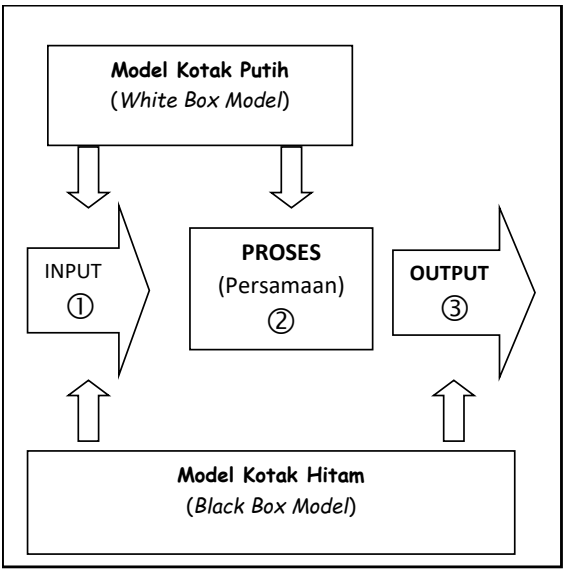
- **Konstanta** sebagai bagian dari model yang mempunyai nilai tetap, misalkan $\pi$ = 3.14 dan seterusnya.
- **Model matematis** merupakan kumpulan dari persamaan, variabel dan konstanta

Hubungan antara model matematis atau persamaan-persamaan matematis dengan variabel sebagai bagian *input* dan *output* dapat dilihat pada gambar 1.3.




Gambar 1.3: Hubungan input – output dalam suatu model

Berdasarkan pada gambar 1.3, apabila model matematis dari sistem proses beserta penyelesaian matematis (nomor 2) sudah dapat dijalankan dengan baik, maka dengan memasukan nilai tertentu pada *input* (nomor 1) sebagai bagian dari stimulus maka akan didapat respons sebagai bagian dari *output* (nomor 3). Sebagaimana yang telah didefinisikan diatas dengan mengubah-ubah nilai maka bagian tersebut merupakan variabel. Variabel yang dapat dengan bebas nilainya diubah-ubah (nomor 1) disebut dengan variabel bebas (*independent variable*). Sedangkan hasil perhitungan matematis maka akan didapat respons juga sebagai variabel tetapi nilainya tergantung dari nilai yang dimasukkan dari variabel bebas, maka variabel keluaran atau respons tersebut dinamai dengan variabel tak bebas (*dependent variable*). Model yang mensimulasikan nilai input (1) pada proses (2) berdasarkan pendekatan teori yang relevan untuk mendapatkan respon (3), atau **[1 + 2 → 3]**, disebut dengan **model kotak putih** (*white box model*).




Gambar 1.4: Perbedaan antara model kotak putih dan hitam

Akan tetapi, ada kalanya, model matematis berdasarkan pada teori penunjang belum ada, maka dapat dilakukan suatu eksperimen sehingga didapat sekumpulan data dari stimulus atau eksperimental variabel (nomor 1) dengan sekumpulan data respon atau behavior variabel (nomor 3), sehingga akan didapat persamaan empiris yang menggambarkan karakteristik proses (nomor 2) atau **[1 + 3 → 2]**. Model ini disebut dengan **model kotak hitam** (*black box model*). Metoda numeris untuk menyelesaikan model tersebut menggunakan metoda pencocokan kurva (*curve fitting*). Selain itu ada pula metoda interpolasi dan ekstrapolasi apabila digunakan untuk mencari data yang tidak tersedia dari hasil percobaan tanpa harus mengetahui bentuk persamaannya. Pencocokan kurva dapat juga digunakan untuk mencari nilai parameter-parameter dari sistem proses. Untuk memperjelas kedua perbedaan tersebut dapat dilihat blok diagram pada gambar 1.4. Hubungan diantara sebagaimana yang

dilihat pada gambar 1.3 maupun gambar 1.4 merupakan hubungan sebab-akibat.

Kompleksnya permasalahan dalam suatu sistem proses, akan mempengaruhi kompleksitas dalam persamaan yang dibentuk. Bentuk persamaan yang relatif sederhana disebut dengan persamaan linear dan sistem modelnya disebut dengan model linear, dilain pihak disebut dengan model non linear. Linearsitas dari persamaan tergantung dari orde (pangkat) pada variabel bebasnya. Apabila orde (pangkat) dari variabel bebasnya satu, maka disebut linear, sebaliknya jika ordenya tidak sama dengan satu disebut dengan non linear.

## 3. Hasil dan Diskusi

Pada bagian ini akan dibahas hasil dan diskusi dari beberapa sistem blok yang merupakan tahapan dari bagian sebelumnya dari suatu sistem. Pada setiap sub bab akan dipresentasikan mulai dari model matematis, script dari Scilab dan pada eksekusi hasil sebagai bagian dari simulasi.

### 3.1. Reaktor Bioproses

#### 3.1.1. Pendekatan model matematis

Pendekatan matematis model proses biologi cukup kompleks, seperti pertumbuhan, predasi, mortalitas, imigrasi, emigrasi pada berbagai macam substat pada biota. Komplesitas model akan benar, jika lebih adari satu nutrient yang berkontribusi pada pertumbuhan *cell*. Pada bagian ini akan dikembangkan model matematis sederhana di turunkan dari persamaan kinetika reaksi pertumbuhan dari biomassa atau *cell* dengan mengkonsumsi nutrient atau substrat menjadi sel baru ditambah produk.

*Cell* (biomassa) + Substrat (*nutrient*) → *Cell* berbiak + Produk [1.1]

Persamaan matematika kinetika dari pertumbuhan cell maupun substrat dikembangkan berdasarkan pada hasil pencocokan data eksperimen. Berikut merupakan beberapa kinetika model:

### 3.1.2. Kecepatan pertumbuhan dari Cell (biomassa) (Fogler, 2006)

$$r_g = \mu . C_c \qquad [1.2]$$

Dimana:

$r_g$ = kecepatan pertumbuhan cell (g/dm$^3$.s)

$\mu$ = kecepatan pertumbuhan spesifik (s$^{-1}$)

$C_c$ = konsentrasi cell (g/dm$^3$)

Koefisien kecepatan pertumbuhan spesifik ($\mu$) biasanya tidak konstant. Hubungan antara koefisien kecepatan pertumbuhan spesifik (μ) dengan konsentrasi substrat $C_S$ sebagai sumber nutrisi cell sudah banyak dikembangkan (Fogler, 2006), (Alt and Markov, 2012), persamaan umum adalah (a) *Monod* dan (b) Andrews - *Substrat Inhibisi*.

**a) Monod**

Koefisien kecepatan pertumbuhan bentuknya hyperbolik.

$$\mu = \frac{\mu_{\max} . C_S}{k_m + C_S} \qquad [1.3]$$

**b) Andrews, Substrat Inhibisi**

Pada beberapa sistem, terbentuknya produk akan menghambat – inhibisi dari kecepatan pertumbuhan cell. Contohnya proses fermentasi glukosa yang memproduksi

ethanol, dimana produk ethanol ini akan menghambat (Esfahanian *et al.*, 2016).

Untuk substrat inhibisi, persamaannya adalah (Bequette, 1998):

$$\mu = \frac{\mu_{\max}.C_S}{k_m + C_S + k_1.C_s^2} \qquad [1.4]$$

Apabila $k_1$=0 (sangat kecil), maka persamaan Andrews-substrat inhibisi (1.4) sama dengan persamaan Monod (1.3).

**c) Monod, produk Inhibisi** (Fogler, 2006)

Pada proses fermentasi glukose memproduksi ethanol, dimana produksi ethanol ini akan menghambat (inhibisi) produksinya. Salah satu persamaan yang dikembangkan adalah (modifikasi persamaan):

$$\mu_p = \left(1 - \frac{C_p}{C_p^*}\right)^n \frac{\mu_{\max}.C_S}{k_m + C_S} \qquad [1.5]$$

Parameter inhibisi untuk fermentasi glokosa menjadi ethanol (Fogler, 2006) adalah $n$ = 0.5 dan $C_P^*$=*93 g/dm*$^3$.

Untuk mendapatkan nilai parameter tersebut diperlukan data hubungan antara $C_S$ dan $\mu$, yang di dapat dari pencocokan data eksperimen laboratorium atau lapangan (biasanya disebut dengan *wet-lab*). Penyelesaian matematis secara numerik dari sistem pencocokan kurva data eksperimen perlu dikembangkan lebih sederhana, khususnya untuk model persamaan yang non-linear. Yang paling sederhana adalah linearisasi sistem persamaan dahulu, kemudian dilakukan regresi linear. Akan tetapi tidak semua persamaan dapat dilinearisasikan. Pada tulisan ini

digunakan teknik optimasi dengan menggunakan metoda least-square.

### 3.1.3. Stoichiometri

Persamaan stoichiometry di simplifikasi hanya satu nutrien (substrat) pada medium. Dengan menggunakan persamaan [1.1], maka bentuk tersebut dapat disederhanakan :

*Cell* (biomassa) + Substrat (*nutrient*) → *Cell* berbiak + Produk [1.1]

Untuk mendapatkan hubungan substrat yang dikonsumsi, cell baru yang terbentuk (berbiak) dan produk, maka didefinisikan parameter baru yaitu YIELD, *Y*.

$$Y_{C/S} = \frac{Massa\ cell\ diproduksi}{Massa\ substrat\ dikonsumsi} = -\frac{\Delta C_C}{\Delta C_S} \qquad [1.6]$$

$$Y_{P/C} = \frac{Massa\ produk\ terbentuk}{Massa\ cell\ baru\ terbentuk} = \frac{\Delta C_P}{\Delta C_C} \qquad [1.7]$$

Sehingga persamaan [1.1] dapat diubah menjadi :

$$S \xrightarrow{cell} Y_{C/S} \times C + Y_{P/S} \times P \qquad [1.8]$$

### 3.1.4. Neraca Massa

Untuk menghitung pengaruh dari pada Volume, kecepatan aliran dari Reaktor, maka perlu dibuat persamaan neraca massa. Secara umum, neraca massa total, dapat dinyatakan sebagai berikut :

$$Akumulasi = kecmasuk - keckeluar \pm reaksi \qquad [1.9]$$

Neraca komponen :

**Neraca massa cell** :

Kec. Akumulasi cell = kec. Aliran Cell masuk – kec. Aliran Cell keluar + kec. Pembentukan cell

$$V\frac{dC_c}{dt} = F_{in}C_{Co} - F_{out}C_C + \left(r_g - r_d\right)V \qquad [1.10]$$

Dimana :

$r_d = k_d C_c$ kecepatan kematian cell

**Neraca massa substrat** :

$$V\frac{dC_S}{dt} = F_{in}C_{So} - F_{out}C_S + r_s\,V \qquad [1.11]$$

$$V\frac{dC_S}{dt} = F_{in}C_{So} - F_{out}C_S + Y_{S/C}.(-r_g)\,V \qquad [1.12]$$

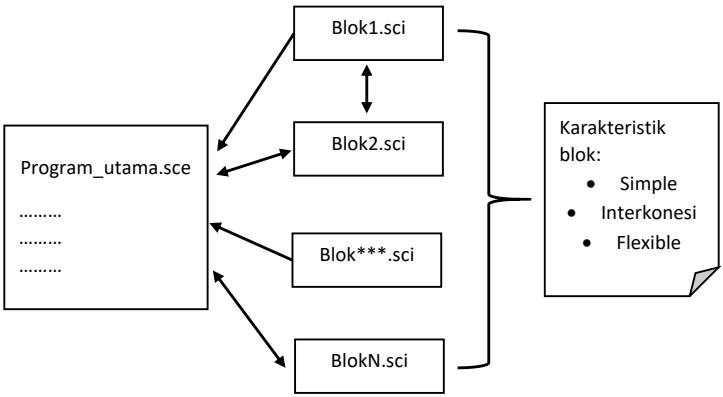
**Neraca massa Produk :**

$$V\frac{dC_P}{dt} = F_{in}C_{Po} - F_{out}C_P + Y_{P/C}.(r_g)\,V \qquad [1.13]$$

Persamaan diferensial harus diselesaikan secara simultan, dan permasalahan dari sistem tergantung dari jenis reaktor. Pada tulisan ini akan dikembangkan dua macam reaktor, yaitu reaktor Batch dan Reaktor Aliran Tangki Berpengaduk (RATB – CSTR *Continous Stirred Tank Reactor*).

Hasil dari tulisan ini berupa kode program komputer dengan menggunakan bahasa pemrogram Scilab 5.2.2, selanjutnya eksekusi program di visualisasikan dalam bentuk gambar. Visualisasi gambar sebagai bagian dari karakteristik hasil dari system bioreactor. Selain itu visualisasi merupakan validasi daripada program dan matematika model. Simulasi dilaksanakan

dengan menggubah-ubah variable manipulasi, maka karakteristik dari system akan lebih mudah untuk di analysis. Program dibuat dalam bentuk blok-blok yang bresifat simple, flexible dan interkoneksi, artinya program dibuat sesimple mungkin sehingga mudah dibaca seperti membaca rumus yang ringkas, flexible artinya bisa digunakan untuk berbagai keadaan dan interkoneksi artinya dapat dikoneksikan antara blok maupun di dalam blok sehingga dapat terbentuk sistem yang lebih komplek sebagai gabungan dari blok. Tiap blok ditulis dengan ekstensi *.sci sedangkan program utama dengan epkstensi *sce. Hal ii dapat dilihat pada gambar 1.5.




Gambar 1.5 : Hubungan sistem blok (ekstensi *.sci) dengan program utama (ekstensi *.sce)

### 3.1.5. Menentukan paramater kecepatan berdasarkan pencocokan data eksperimen

Untuk mendapatkan nilai parameter pada persamaan Monod, maka diperlukan data eksperimen atau lapangan tentang hubungan antara kecepatan pertumbuhan spesifik ($\mu$) dengan konsentrasi substrat, sebagaimana pada tabel 1.1 berikut.

Tabel 1.1. Hubungan antara koefisien pertumbuhan spesifik ($\mu$) dan konsentrasi substrat $C_S$ (Bequette, 1998).

| $\mu$ | $hr^{-1}$ | 0.25 | 0.31 | 0.36 | 0.43 | 0.45 | 0.47 | 0.50 | 0.52 |
|---|---|---|---|---|---|---|---|---|---|
| $C_S$ | g/liter | 0.1 | 0.15 | 0.25 | 0.50 | 0.75 | 1.00 | 1.50 | 3.00 |

Metode penyelesaian, digunakan dua macam macam, pertama dengan linearisasi persamaan, kemudian diselesaikan dengan regresi linear ; kedua dengan menggunakan metoda least-square yang tidak perlu linearisasi persamaan.

a. Linearisasi persamaan Monod : $\mu = \frac{\mu_{\max}.C_S}{k_m + C_S}$

Dilakukan pembalikan : $\frac{1}{\mu} = \frac{k_m + C_S}{\mu_{\max}.C_S}$

Diubah menjadi bentuk linear : $\underset{y}{\frac{1}{\mu}} = \underset{a}{\frac{k_m}{\mu_{\max}}} \underset{x}{\frac{1}{C_S}} + \underset{b}{\frac{1}{\mu_{\max}}}$

**Program1.1 : Program utama penentuan parameter dengan regresi linear (reglin)**

D:\1_SbSci\Monodreglin.sce

```
//regresi linear
clc
CS=[0.1;0.15;0.25;0.5;0.75;1;1.5;3]'
MU=[0.25;0.31;0.36;0.43;0.45;0.47;0.50;0.52]'
x=1 ./CS      //perlu spasi
y=1 ./MU    //perlu spasi
[a,b]=reglin(x,y)
Mumax=1/b;
km=a*Mumax
Muhit=Mumax.*CS./(km+CS)
```

```
disp(Mumax,km)
plot(CS,MU,'*')
plot(CS,Muhit,'-')
```

b. Metoda least-square tanpa perlu dilakukan linearisasi :

**Program1.2 : Program utama, penentuan parameter dengan metoda leastsquare**

D:\1_SbSci\Monodleastsq.sce

```
clc, clear
exec('D:\1_SbSci\errmonod.sci');
exec('D:\1_SbSci\monod1.sci');
//param(1)=mumax; param(2)=km - Monod
CS=[0.1;0.15;0.25;0.5;0.75;1;1.5;3]
MU=[0.25;0.31;0.36;0.43;0.45;0.47;0.50;0.52]
param_init=[0.1;0.1]

[f,paraOpt]=leastsq(list(errmonod,CS,MU),param_init)
disp('Mumax dan km - parameter Monod')
disp(paraOpt)
disp(f)
MUhit=monod1(CS,paraOpt)
plot(CS,MUhit,'-')
plot(CS,MU,'*')
xtitle('Perhitungan parameter Monod - least-
square','Konsentrasi Substrat','Mu - Monod')
```

**Program1.3 : Blok – error monod**

D:\1_SbSci\errmonod.sci

```
function e=errmonod(param, CS, MU, m)
   e=abs(monod1(CS,param)-MU)
endfunction
```

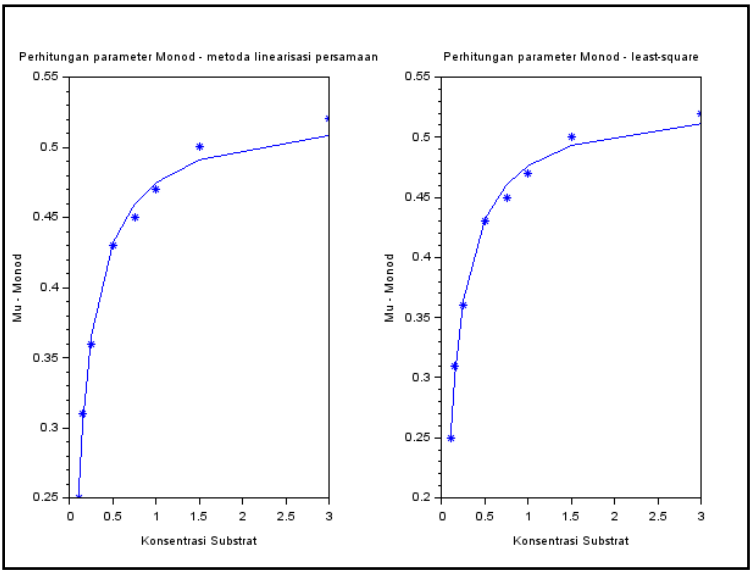
**Program1.4 : Blok - monod**

D:\1_SbSci\monod1.sci

```
function mum=monod1(c, param)
//param(1)=mumax; param(2)=km
   mum=(param(1).*c)./(param(2)+c)
endfunction
```

c. Hasil paramater yang didapat :

| Hasil / Metoda | $\mu_{MAX}$ | $k_m$ | Keterangan hasil |
|---|---|---|---|
| Regresi liniear -**reglin** | 0.109979 | 0.527104 | • Persamaan non linear, harus dilinearkan dahulu.<br>• Bersifat khusus |
| Least square - **leastsq** | 0.113684 | 0.530586 | • Persamaan non-linear tidak perlu dilinearkan<br>• Dapat digeneralisasikan |




Gambar 1.6. Visualisasi data dan garis hasil perhitungan antara metoda **reglin** dan **leastsq**.

d. Dengan menggunakan data yang sama, akan dihitung parameter substrat inhibisi

Pada sub bagian ini akan dihitung nilai parameter dengan menggunakan persamaan substrat inhibisi untuk data yang sama, dengan menggunakan metoda least-square.

**Program1.5 : menghitung parameter untuk persamaan monod dan substrat inhibisi**

D:\1_SbSci\Monodleastsq_Mon_Inhib.sce

```
clc, clear
exec('D:\1_SbSci\errmonod.sci');
exec('D:\1_SbSci\monod1.sci');
exec('D:\1_SbSci\subInhib.sci');
exec('D:\1_SbSci\errInhib.sci');
//Data
```

```
CS=[0.1;0.15;0.25;0.5;0.75;1;1.5;3]
MU=[0.25;0.31;0.36;0.43;0.45;0.47;0.50;0.52]
param_init=[0.1;0.1]

[f,paraOpt]=leastsq(list(errmonod,CS,MU),param_init)
disp('Mumax dan km - parameter Monod')
disp(paraOpt)
disp(f)

param_init1=[0.1;0.1;0.1]
[f1,paraOpt1]=leastsq(list(errInhib,CS,MU),param_init1)

disp('Mumax, km dan k1 - parameter Subtrat Inhibisi')
disp(paraOpt1)
disp(f1)

MUMohit=monod1(CS,paraOpt)
plot(CS,MUMohit,'-')
plot(CS,MU,'*')
//xtitle('Perhitungan parameter Monod - least-
square','Konsentrasi Substrat','Mu - Monod')
MUINhit=subInhib(CS,paraOpt1)
plot(CS,MUINhit,'--')
xtitle('Perbandingan paramter Monod & Substrat
Inhibisi','Konsentrasi Substrat','Mu - Monod (-) vs Subst In (--) vs
Data (*)')
```

**Program1.6 : Blok – Substrat Inhibisi**

D:\1_SbSci\subInhib.sci

```
function mus=subInhib(c, param)
  //param(1)=mumax; param(2)=km; param(3)=k1
  mus=(param(1).*c)./(param(2)+c+param(3).*c.^2)
endfunction
```

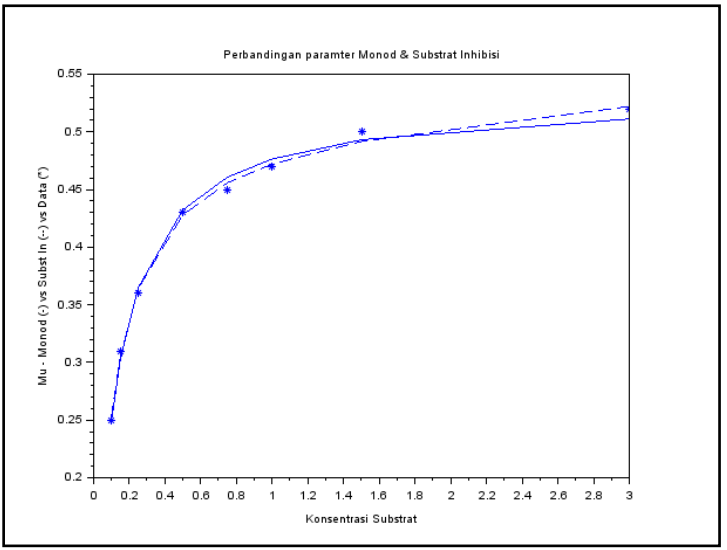
**Program1.7 : Blok – error Substrat Inhibisi**

D:\1_SbSci\errInhib.sci

```
function e=errInhib(param, CS, MU, m)
  e=abs(subInhib(CS,param)-MU)
endfunction
```

Hasil nilai parameter dari metoda reglin dan leastsq

| Hasil / Metoda | $\mu_{MAX}$ | $k_m$ | $k_1$ | Bentuk persamaan |
|---|---|---|---|---|
| Regresi liniear - **reglin** | 0.109979 | 0.527104 | 0 | $\mu = \frac{\mu_{\max}.C_S}{k_m + C_S}$ |
| Least square - **leastsq** | 0.113684 | 0.530586 | 0 | |
| Least square - **leastsq** | 0.1033 | 0.5127 | - 0.0174 | $\mu = \frac{\mu_{\max}.C_S}{k_m + C_S + k_1.C_s^2}$ |




Gambar 1.7. Visualisasi data dan garis hasil perhitungan antara metoda **leastsq** untuk persamaan Monod (-) dan persamaan substrat inhibisi (--).

e. Visualisasi persamaan Monod dan Substrat Inhibisi dengan diketahui parameter

Pada bagian ini, nilai parameter kedua persamaan sudah diketahui, kemudian akan dilihat karakteristik dari kedua model persamaan.

Paramater Monod : $\mu_{max}$ = 0.53 $hr^{-1}$ ; $k_m$ = 0.12 g/liter

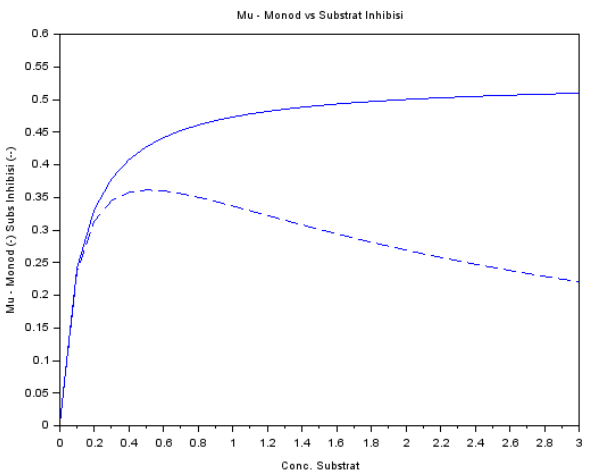
Paramater Substrat Inhibisi : $\mu_{max}$ = 0.53 $hr^{-1}$ ; $k_m$ = 0.12 g/liter ; $k_1$ = 0.4545 liter/g

**Program1.8 : Program utama – membandingkan Monod vs Substrat Inhibisi dengan diketahui parameter.**

D:\1_SbSci\Mon_Inhib_curva.sce

```
clc, clear
exec('D:\1_SbSci\monod1.sci');
```

```
exec('D:\1_SbSci\subInhib.sci');

//param(1)=mumax; param(2)=km; param(3)=k1
paramonod=[0.53;0.12];
parasub=[0.53;0.12;0.4545];
CS=0:0.1:3
Mum=monod1(CS,paramonod);
Muhib=subInhib(CS,parasub);
plot(CS,Mum,'-')
plot(CS,Muhib,'--')
xtitle('Mu - Monod vs Substrat Inhibisi','Conc. Substrat','Mu -
Monod (-) Subs Inhibisi (--)')
```




Gambar 1.8. Visualisasi karakteristik persamaan Monod dan Andrew (Substrat Inhibisi).

Pada gambar 1.8, terlihat bahwa persamaan Monod tidak melihat adanya faktor inhibisi pertumbuhan dengan adanya

peningkatan nutrisi (substrat), Monod melihat semakin banyak substrat maka kecepatan pertumbuhan spesifik juga akan semakin naik, akan tetapi Andrew melihat, dengan substrat semakin tinggi akan menjadi racun bagi cell, sehingga terjadi penurunan kecepatan pertumbuhan spesifik.

Pada sub-bagian ini, metoda untuk pencocokan data eksperimen dengan menggunakan metoda least-square lebih sederhana dibandingkan dengan metoda regresi linear, hal ini pertama pada metoda least-square tidak perlu dilakukan linearisasi persamaan. Kedua, pada metoda least-square dapat dibuat umum (general), dengan demikian persamaan dasarnya (Monod atau Substrat Inhibisi) dapat digunakan untuk blok yang lain.

### 3.1.6. Simulasi pada sistem reaktor kontinyu berpengaduk

Pada bagian ini akan disimulasikan hasil dari **program1.9**. Kecepatan aliran pertumbuhan spesifik (μ) menggunakan persamaan Substrat Inhibisi. Simulasi akan dilakukan dengan membedakan kondisi awal, kasus 1 : Konsentrasi Cell (Biomass) = 0.75 g/liter dan Konsentrasi Substrat =2 g/liter ; kasus 2 : konsentrasi cell (biomass) dan substrat sama yaitu 1 g/liter. Hasil dari keduanya di visualisasikan dalam bentuk gambar. Program9 disimulasikan dengan variasi dari Kecepatan pengenceran (D=F/V, $hr^{-1}$), terbagi menjadi tiga kasus :

Kasus1 : Kecepatan pengenceran rendah, D=0.15 $hr^{-1}$.
Kasus2 : Kecepatan pengenceran sedang, D=0.3 $hr^{-1}$.
Kasus3 : Kecepatan pengenceran tinggi, D=0.45 $hr^{-1}$.

**Program1.9 : Program Utama penyelesaian Bioreaktor tipe Aliran Tangki berpengaduk**

D:\1_SbSci\latih10_bioreakCSTR.sce

```
clc,clear
exec('D:\1_Sbsci\bioreak_CSTR.sci');
exec('D:\1_SbSci\subInhib.sci');
//D,mumax,Y,km,sf,k1
D=input('dilution rate= ')
param=[D;0.53;0.4;0.12;4;0.4545];

for i=1:2
if i==1 then cd0=[0.75;2]; end
if i==2 then cd0=[1;1]; end
t0=[0;0];
t=0:0.1:20;
cd=ode(cd0,t0,t,list(bioreak_CSTR,param))
cd1=cd(1,:)';
cd2=cd(2,:)';
t=t';
subplot(1,2,i)
plot2d([t t],[cd1 cd2],[1 -1])
xtitle('Bioreactor simulation - S Budi Sasongko','Time','Biomas
Conc. (-) dan Substrat Conc. (+)')
end
```

**Program1.10 : blok CSTR bio**

D:\1_SbSci\bioreak_CSTR.sci

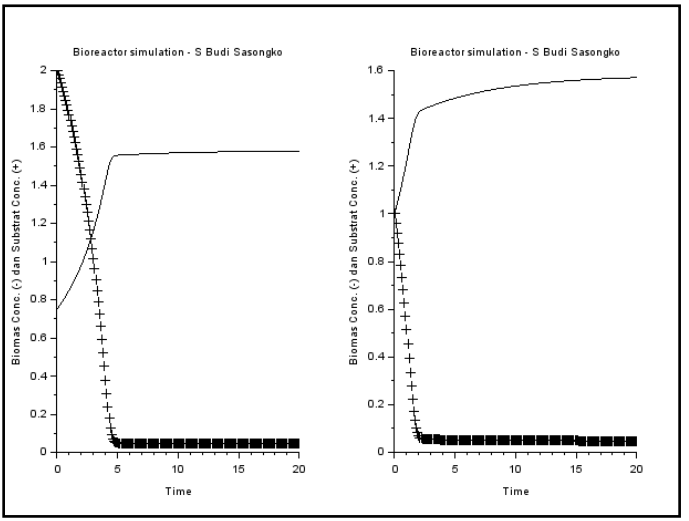
```
function cdot=bioreak_CSTR(t, c, param)
//Andrew - Substrat Inhibisi
   D=param(1);
```

```
   mumax=param(2);
   Y=param(3);
   km=param(4);
   sf=param(5);
   k1=param(6);
paramSI=[mumax,km,k1]

  mu=subInhib(c(2),paramSI)
//persamaan dinamis

cdot(1)=(mu-D)*c(1);
cdot(2)=(sf-c(2))*D-mu*c(1)/Y;
endfunction
```

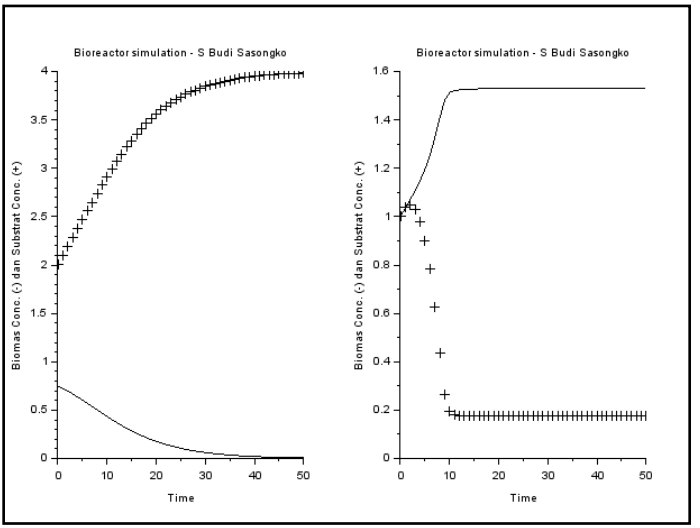
**Program1.11 : Blok substrat Inhibisi = program6**

D:\1_SbSci\subInhib.sci

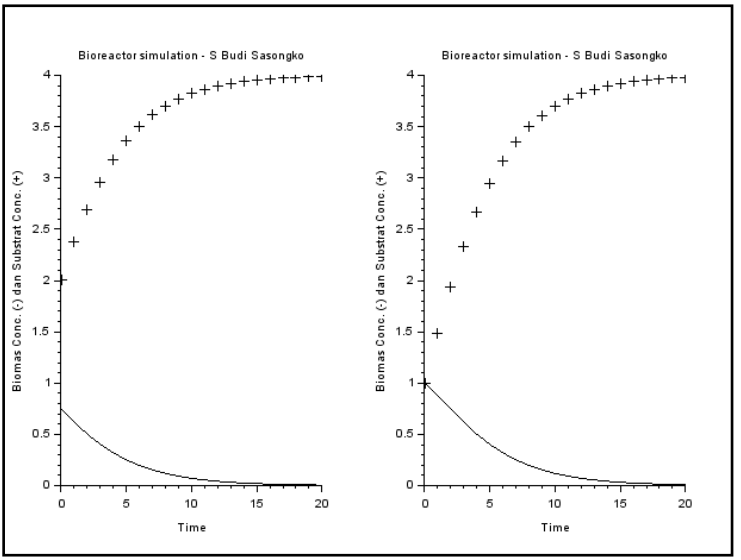
```
function mus=subInhib(c, param)
   //param(1)=mumax; param(2)=km; param(3)=k1
   mus=(param(1).*c)./(param(2)+c+param(3).*c.^2)
endfunction
```




Gambar 1.9 : Dinamika karakteristik konsentrasi Cell-Biomass (-) dan Substrat (--) dengan kondisi awal yang berbeda, pada D=0.15 $hr^{-1}$ (rendah)




Gambar 1.10 : Dinamika karakteristik konsentrasi Cell-Biomass (-) dan Substrat (--) dengan kondisi awal yang berbeda, pada D=0.30 $hr^{-1}$ (sedang)




Gambar 1.11 : Dinamika karakteristik konsentrasi Cell-Biomass (-) dan Substrat (--) dengan kondisi awal yang berbeda, pada D=0.45 $hr^{-1}$ (tinggi)

Berdasarkan dari hasil simulasi, untuk kecepatan pengenceran yang tinggi, maka terlihat konsentrasi substrat selalu pada posisi yang lebih tinggi dibanding konsentrasi sel (gambar1.11). Disisi lain, untuk kecepatan pengenceran yang rendah, maka konsentrasi substrat akan turun dibawah konsentrasi cell (biomass) sebagaimana ditunjukkan pada gambar 1.9.

### 3.2. Reaktor model konversi tetap.

Pada bagian ini akan dibahas mengenai proses perhitungan untuk reaktor dengan konversi tetap. Sebagai ilustrasi adalah peristiwa oksidasi dari gas methane dan gas ethane. Dimana kedua reaksi tersebut berjalan secara paralel atau

simultan dan konversi dari keduanya tetap. Berikut data-data yang ada :

Reaksi oksidasi berikut :

$CH_4 + 2\ O_2 \rightarrow CO_2 + 2\ H_2O$ $X_1 = 60\%$

$C_2H_6 + 7/2\ O_2 \rightarrow 2\ CO_2 + 3\ H_2O$ $X_2 = 80\%$

Komposisi masuk (% mole) :

12.5% $CH_4$ ; 75% $O_2$ ; 12.5% $C_2H_6$ ;

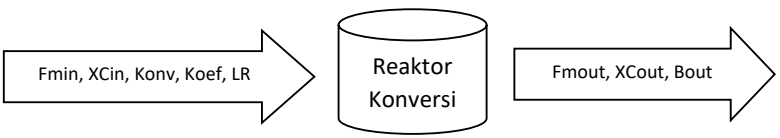
Kecepatan aliran molar masuk : 80 mole/jam

### 3.2.1. Pendekatan model matematis

Bentuk persamaannya secara umum :

$$L_{out,k} = L_{in,k} + \sum_{rx=1}^{NRx} CS_{rx,k}.X_{rx}.L_{in,LR-Rx} \qquad [1.14]$$

Hubungan blok program, variabel input dan variabel output dapat dilihat pada gambar 1.12.




Gambar 1.12. Hubungan antara blok program, parameter input dan output.

**Program1.12 : Blok reaktor konversi**

D:\sbscilab\reaksto.sci

```
1  function [Fmout, XCout, Bout]=reaksto(Fmin, XCin, Konv,
   Koef, LR)
2  //Reaktor Konversi tetap
   //Fm = molar flowrate total = mol/waktu
   //B = molar flowrate komponen = mol/waktu
   //XC = fraksi mol komponen
   //LR: Limiting reaktan, dalam bentuk baris,
   //misal: LR=[1 3]
   //Konv dalam bentuk matrix kolom
3  Bin=Fmin.*XCin;
4    ukr=size(LR);
     jr=ukr(1,2);
5    for i=1:jr
       BLR(1,i)=Bin(1,LR(1,i));
     end
     BLR=BLR';
6  Xblr=Konv.*BLR;
7    Koef=Koef';   //ubah ke bentuk kolom
     Brx=Koef*Xblr;   //flowrate reaksi
8    Bout=Bin'+Brx;
9    Fmout=sum(Bout);
     XCout=Bout./Fmout;
10 endfunction
```

**Program1.13 : Aplikasi program blok reaktor konversi**

Pada program berikut merupakan aplikasi untuk reaksi oksidasi dari methane dan ethane, dengan komposisi dan aliran masuk sebagaimana ilustrasi diatas.

| | |
|---|---|
| 1 | clc; clear<br>*//CH4 O2 C2H6 CO2 H2O Daftar komponen*<br>exec("D:\sbscilab\reaksto.sci"); |
| 2 | Cin1=[0.125 0.75 0.125 0 0];<br>Fin1=80;<br>Koef=[-1 -2 0 1 2; 0 -7/2 -1 2 3]; *//Koefisien stokhiometri*<br>Konv=[0.6; 0.8];<br>LC=[1 3]; |
| 3 | [Fout,Cout, Bout]=reaksto(Fin1,Cin1,Konv,Koef,LC) |
| 4 | disp("Flow rate output ")<br>disp(Fout)<br>disp("Konsentrasi Output:")<br>disp(Cout)<br>disp("Molar ")<br>disp(Bout) |

### 3.3. Properti paket untuk aplikasi gas non-ideal.

Hukum gas ideal dinyatakan dengan hubungan antara Tekanan-Volume-Suhu (PVT) pada kondisi tekanan atmosferik. Pada tekanan yang tinggi, maka persamaan keadaannya (EoS – *Equation of State*) menjadi lebih kompleks. Sebagai ilustrasi adalah **gas Ammonia**, pada tekanan tinggi, persamaan keadaannya menggunakan bentuk **van der Waals**, dalam bentuk :

$$\left(P+\frac{a}{V^2}\right)(V-b)=RT \qquad [1.15]$$

Dimana :

$$a=\frac{27}{64}\left(\frac{R^2T_c^2}{P_c}\right) \qquad [1.16]$$

$$b = \frac{RT_c}{8P_c} \qquad [1.17]$$

Variabel diatas didefinisikan sebagai berikut :
*P* = tekanan dalam atm
*V* = volume molar dalam liter/g-mol
*T* = temperatur dalam K
*R* = konstanta gas (*R* = 0.08206 atm.liter/g-mol.K)
$T_c$ = Temperatur kritis (405.5 K untuk Ammonia)
$P_c$ = Tekanan kritis (111.3 atm untuk Ammonia)

Tekanan reduksi :

$$P_r = \frac{P}{P_c} \qquad [1.18]$$

Faktor kompresibilitas :

$$Z = \frac{PV}{RT} \qquad [1.19]$$

Permasalahan yang akan diselesaikan :

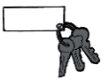
- Menentukan volume molar (V) pada tekanan tinggi (56 atm), suhu 450 K
- Akan dilihat pengaruh tekanan tinggi terhadap faktor kompresibilitas (Z) atau *Z=f* (*P*).

### 3.3.1 Penyelesaian model matematis dengan metoda numerik

Pada penyelesaian akar persamaan atau persamaan non linear, menggunakan metoda numerik. Penyelesaian dengan metoda analitis, relatif lebih rumit dan penyelesaian terbatas

dalam generalisasi. Untuk pembuktian validitas dari hasil perhitungan, nilai hasil dimasukan kembali pada persamaan, dengan hasil akhir sama dengan (mendekati) nol.

**Syntax** penyelesaian Sistem Non Linear menggunakan Scilab untuk akar-akar persamaan atau fungsi sama dengan nol:

| | |
|---|---|
|  | [y]=**fsolve**(x0,fce)<br>x0=nilai awal (tebakan)<br>fce=fungsi external (fungsi akar pesamaannya) |

Untuk penyelesaian persamaan diatas, maka pada persamaan van der Waals (persamaan 1.15) diatas diubah $f(V) \approx 0$.

$$f(V) = \left( P + \frac{a}{V^2} \right)(V - b) - RT \approx 0 \qquad [1.20]$$


**Program1.14 : Blok van der Waals**

```
function f=vdWaals(V, P, T, Pc, Tc)

//Menghitung volume molar pers. van der Waals unit: liter/gmol
//P tekanan unit atm
//T temperatur unit K
//Pc dan Tc tekanan dan temperature kritis
   R=0.08206;  //atm.liter/g-mol.K
   a=(27/64)*((R^2)*(Tc^2)/Pc)
```

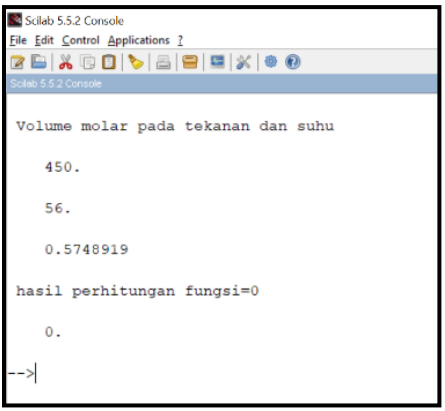
| | |
|---|---|
| | b=(R*Tc)/(8*Pc)<br><br>//persamaan utama:<br>f=(P+(a/V.^2))*(V-b)-R*T<br>endfunction |

| | |
|---|---|
|  | **Penerapan blok:**<br>Volume molar (V) pada tekanan tinggi (56 atm), suhu 450K |

**Program1.15: Aplikasi program blok van der Waals**

| | |
|---|---|
| 1 | clc; clear |
| 2 | *//penyelesaian persamaan van der Waals*<br>*//ammonia* |
| 3 | exec(' D:\sbscilab\vdWaals.sci'); |
| 4 | P=56; *//atm*<br>T=450; *//K*<br>Tc=405.5 *//K - ammonia*<br>Pc=111.3 *//atm - ammonia* |
| 5 | vh=fsolve(1,vdWaals) |
| 6 | disp(P,T,'Volume molar pada tekanan dan suhu ')<br>disp(vh) |
| 7 | *//Prove - pembuktian f(v)==0*<br>fnol=vdWaals(vh,P,T,Pc,Tc) |
| 8 | disp('hasil perhitungan fungsi=0')<br>disp(fnol) |

| | |
|---|---|
|  | **Hasil (Produk):**<br>Volume molar (V) pada tekanan tinggi (56 atm), suhu 450K |




Gambar 1.13. Hasil eksekusi Volume molar Amonia pada tekanan 56 atm, 450 K.

Pada gambar 1.13 terlihat hasil dari perhitungan volume molar pada tekanan 56 atm dan suhu 450 K adalah sebesar 0.5748919 liter. Pada perhitungan tersebut nilai $f(V) = 0$ sebagaimana persamaan 1.20.

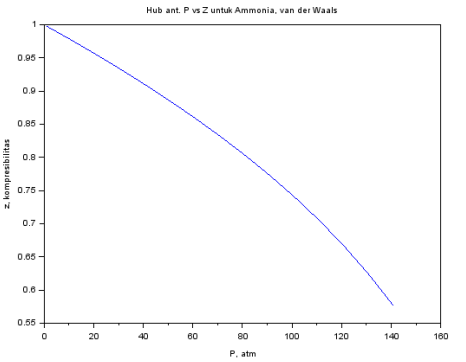
| | |
|---|---|
|  | **Penerapan blok:**<br>Pengaruh tekanan tinggi terhadap faktor kompresibilitas (Z) atau *Z=f* (*P*) |

**Program1.16: Aplikasi program blok van der Waals pada kompresibilitas fungsi tekanan**

```
1   Ps=[1:10:150]
    vmol=[]
    zs=[]
    n=size(Ps)
    nk=n(1,2)
2   T=450;     //K
    Tc=405.5   //K - ammonia
```

| | |
|---|---|
| | Pc=111.3 *//atm - ammonia*<br>R=0.08206; *//atm.liter/g-mol.K* |
| 3<br>4<br>5<br>6<br>7<br>8 | for i=1:nk<br>P=Ps(i)<br>vm=**fsolve**(1,vdWaals3)<br>vmol(i)=vm<br>zs(i)=P*vm/(R*T)<br>end |
| 9 | plot(Ps,zs)<br>xtitle("Hub ant. P vs Z untuk Ammonia, van der Waals","P, atm","z, kompresibilitas") |

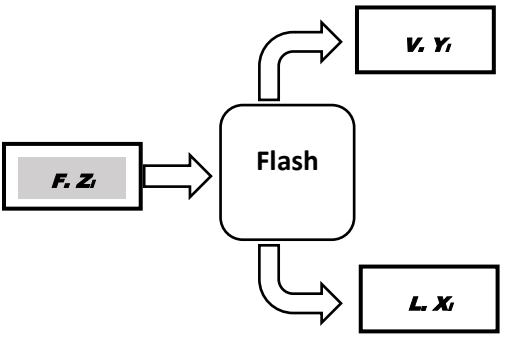
Hasil eksekusi dari program 1.16 sebagaimana ditunjukkan pada gambar 1.14. Pada gambar sumbu x merupakan variasi tekanan mulai dari 1 atm sampai dengan 140 atm, sedangkan sumbu y merupakan faktor kompresibilitas *z* dari gas Amonia tersebut. Pada gambar terlihat, pada tekanan 1 atm nilai faktor kompresibilitas sama dengan satu, hal ini sesuai dengan kondisi gas ideal, akan tetapi semakin tinggi tekanan, maka faktor kompresibilitas akan semakin turun.




Gambar 1.14. Korelasi Tekanan dengan Z (kompresibiltas) dari Ammonia pada suhu 450 K.

### 3.4. Kesetimbangan fase Uap-Cair multi komponen pada flash destilasi.

Pada bagian ini akan dibahas mengenai sistem blok untuk kesetimbangan multi komponen uap-cair pada flash destilasi. Untuk menyederhanakan masalah sistemnya merupakan isotermal, sehingga nilai $K_i$ kesetimbangan uap-cair pada komponen *i* diketahui tetap. Hubungan antara umpan masuk, Uap dan Liquid dapat dilihat pada gambar 1.15




Gambar 1.15. Flash destilasi, kesetimbangan satu tahap.

#### 3.4.1. Model matematik

$$F = L + V \qquad [1.21]$$

$$z_i F = x_i L + y_i V \qquad [1.22]$$

$$K_i = y_i / x_i \qquad \text{Konstanta kesetimbangan Uap/Cair} \qquad [1.23]$$

$$\sum_{i=1}^{n} x_i = \sum_{i=1}^{n} y_i = 1 \qquad [1.24]$$

Kombinasi antara persamaan [1.21], [1.22], dan [1.23] akan didapat persamaan:

$$\sum_{i=1}^{n} \frac{z_i K_i F}{V(K_i - 1) + F} = 1 \qquad [1.25]$$

Dari persamaan [1.25], diselesaikan dengan metoda numerik akar persamaan:

$$f(V) = \sum_{i=1}^{n} \frac{z_i K_i F}{V(K_i - 1) + F} - 1 \approx 0 \quad \rightarrow \text{dapat nilai } V \qquad [1.26]$$

- Setelah didapat V, maka nilai *L* akan didapat (dari 1.26)
- Korelasi persamaan [1.22] dan [1.23] akan dicari komposisi $y_i$.

$$z_i F = (y_i / K_i) L + y_i V$$
$$z_i F = y_i \{(L/K_i) + V\}$$
$$y_i = \frac{z_i . F}{(L/K_i) + V} \qquad [1.27]$$

- Nilai $x_i$ didapat dari persamaan [1.23],

$$x_i = \frac{y_i}{K_i} \qquad [1.28]$$

Sebagai ilustrasi gas alam yang akan di pisahkan dengan flash (Constantinides, Alkis., Mostoufi, 1999).

Umpan flash destilasi F sebesar 100 mol per jam dari *n* komponen gas alam, pada suhu 48$^{o}$C dan tekanan 11 MPa, dengan data sebagaimana di tunjukkan pada table 1.2 yang merupakan

hubungan antara komponen, komposisi serta nilai *K* komponen pada suhu dan tekanan diketahui.

Tabel 1.2 Komposisi dan $K_i$ komponen dari gas Alam

| Komponen | *i* | $z_i$ | $K_i$ |
|---|---|---|---|
| Methane | 1 | 0.8345 | 3.090 |
| Carbon dioxide | 2 | 0.0046 | 1.650 |
| Ethane | 3 | 0.0381 | 0.720 |
| Propane | 4 | 0.0163 | 0.390 |
| *i*-Butane | 5 | 0.0050 | 0.210 |
| *n*-Butane | 6 | 0.0074 | 0.175 |
| Pentanes | 7 | 0.0287 | 0.093 |
| Hexanes | 8 | 0.0220 | 0.065 |
| Heptane+ | 9 | 0.0434 | 0.036 |
| Total | | 1.0000 | |

Permasalahan akan dicari jumlah dan komposisi keluar flash destilasi berupa uap (Vapor) dan Cair (Liquid).

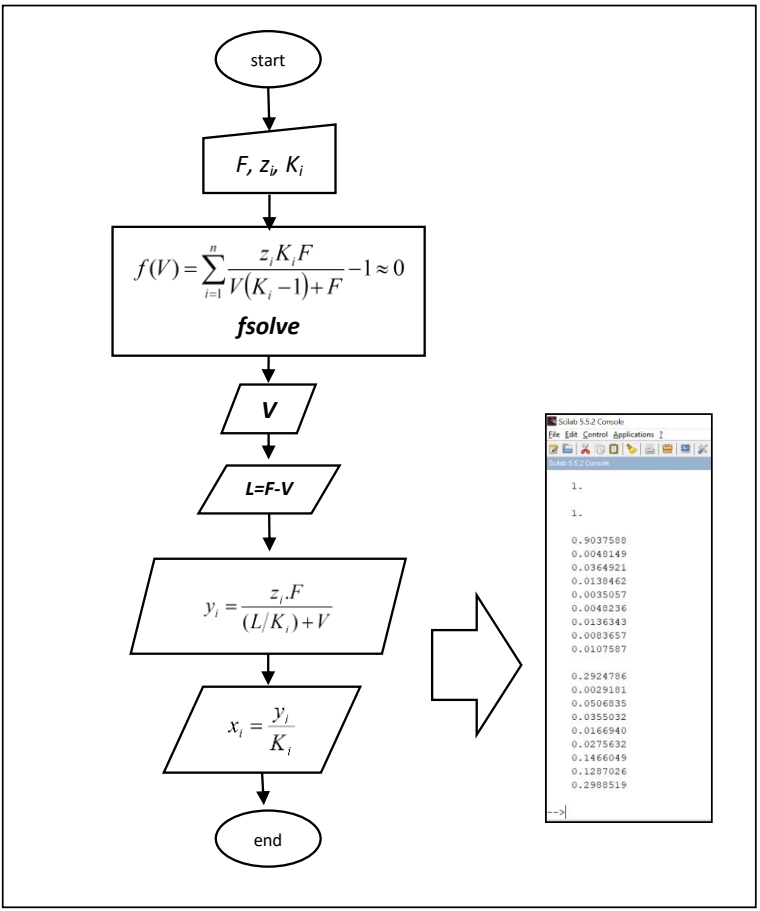
**Program1.17: Blok flash destilasi**


```
function vh=flash(V, F, K, Z)
//Z atau K dalam bentuk vektor baris (row vector)
   nk=size(Z);
   ni=nk(1,2)
   vs=[]
   for i=1:ni
      vs(i)=(F*Z(i)*K(i))/((V*(K(i)-1)+F))
   end
   vh=sum(vs)-1
endfunction
```

**Program1.18 : Aplikasi program blok Flash**

| | |
|---|---|
| 1 | *//flash destilation:*<br>clc, clear<br>**exec**("D:\sbscilab\flash.sci"); |
| 2 | Z=[0.8345,0.0046,0.0381,0.0163,0.005,0.0074,0.0287,0.022,0.0434];<br>K=[3.09,1.65,0.72,0.39,0.21,0.175,0.093,0.065,0.036];<br>F=100<br>*//Mass balace*<br>V=**fsolve**(1,flash)<br>disp(V) |
| 3 | *// mass balance*<br>L=F-V<br>nk=**size**(Z); |
| 4 | *// component mass balance*<br>yhit=[]<br>for i=1:nk(1,2)<br>yhit(i)=(Z(i)*F)/((L/K(i))+V)<br>end |
| 5 | xhit=yhit'./K |
| 6 | xhit=xhit'<br>yc=sum(yhit)<br>disp(yc)<br>xc=sum(xhit)<br>disp(xc)<br>disp(yhit)<br>disp(xhit) |

Program 1.17 merupakan blok flash destilasi sederhana untuk multikomponen sebagai bagian dari persamaan [1.26], selanjutnya persamaan tersebut diselesaiakan dengan metoda numerik akar persamaan dengan sub-rutin **fsolve**, seperti pada bagian 2 pada program 1.18. Tahapan dari penyelesaian dapat dilihat pada gambar 1.16.




Gambar 1.16. Diagram alir penyelesaian blok Flash Destilasi

### 3.5. Aplikasi sistem dengan blok dengan *Recycle*

Pada sub bagian ini akan dibahas mengenai aplikasi dari sistem yang terdiri dari sekumpulan blok-blok. Ilustrasi sederhana sebagai berikut, suatu **reaktor konversi** melakukan proses reaksi, selanjutnya dilakukan pemisahan dengan unit **splitter** keluaran dari unit splitter ini sebagian dikembalikan kembali ke reaktor kinversi. Selanjutnya, digunakan permasalahan sebagaimana pada sub-bab 3.2 tentang Reaktor model konversi. Pada sub bab

tersebut dibahas mengenai reaksi oksidasi dari Methane ($CH_4$) dan Ethane ($C_2H_6$), dengan komposisi masuk (% mole) : 12.5% $CH_4$ ; 75% $O_2$ ; 12.5% $C_2H_6$ , dengan aliran molar masuk : 80 mole/jam. Konversi pada reaksi pertama (oksidasi Methane) 60%, sedangkan reaksi kedua (Oksidasi Ethane) sebesar 80%.

Karena reaksi belum maksimal, maka akan disimulasikan sebagian dari produk reaksi akan dikembalikan sebagian sebagai umpan balik (arus *recycle*) dicampur bersama umpan segar (*Feed*). Oleh karenanya diperlukan **Splitter** untuk memisahkan sebagian hasil dari **Reaktor** sebagai produk akhir dan aliran yang di *recycle*. Pada kasus ini, Splitter berfungsi memisahkan aliran dari satu menjadi dua dengan komposisi yang sama. Kemudian umpan balik (*recycle*) akan dicampur bersama umpan segar pada **Mixer**.

Penyelesaian sistem dengan menggunakan metoda blok, artinya setiap blok dibagian awal diselesaiakan terlebih dahulu, keluaran dari blok tersebut digunakan sebagai input pada blok selanjutnya. Oleh karenanya pada sistem ini pertama perlu dibuat PFD (*Process Flow Diagram*).

Sehingga apabila dibuat PFD, pertama kali adalah unit **Mixer**, hal ini disebabkan pada Mixer merupakan percampuran antara Umpan Segar (*Feed*) dengan aliran umpan balik, keluaran dari **Mixer** akan masuk pada **Reaktor konversi** yang akan melakukan proses reaksi. Selanjutnya keluaran dari **Reaktor** akan masuk pada **Splitter** yang akan membagi dua aliran keluar reaktor, yaitu aliran produk dan aliran yang akan masuk sebagai umpan balik.

**Permasalahannya** adalah nilai yang masuk **mixer** dari aliran umpan balik tidak atau belum diketahui pada pada saat awal.

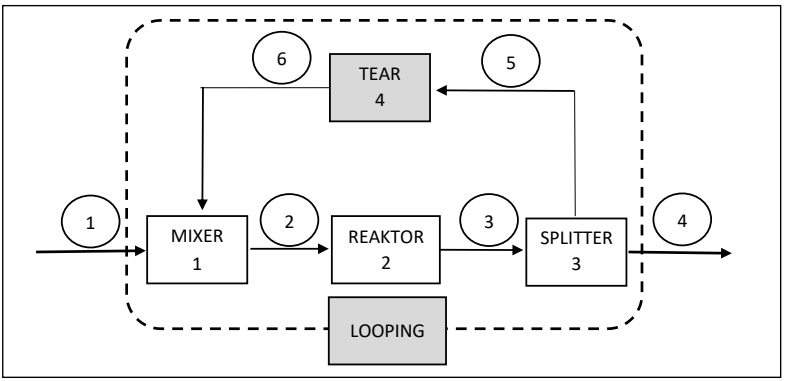
Metoda untuk menyelesaian permasalahan tersebut dengan cara menambah satu unit secara virtual, artinya unit (blok)

tersebut ada secara perhitungan atau virtual, tetapi tidak ada secara fisik. Unit tersebut disebut dengan ***Tear*** artinya pertama kali akan dipisah terlebih dahulu antara aliran yang keluar dari **splitter** sebagai umpan balik dan aliran umpan balik yang akan masuk pada **mixer**. Dengan demikian PFD beserta nomor unit beserta nomor arusnya sebagaimana terlihat pada gambar 1.17. Metode penyelesaian numerik pada unit **Tear** ini adalah metoda yang sederhana yaitu bagi dua. Unit Tear (4) memotong aliran umpan balik dari menjadi dua yaitu aliran lima (5) dan enam (6). Pada kondisi awal semua nilai pada sistem, kecuali aliran satu (1) dibuat sama dengan nol. Perhitungan dimulai dari mixer didapat aliran dua (2), perhitungan unit reaktor didapat aliran tiga (3), selanjutnya perhitungan unit Splitter didapat aliran empat (4) dan lima (5). Metoda perhitungan pada unit **Tear** adalah membagi dua nilai aliran lima (5) dan nilai aliran enam (6). Selanjutnya perhitungan diulang kembali atau ***looping*** sampai didapat nilai antara arus lima (5) dan enam (6) mendekati sama.

Pada gambar 1.17 terlihat garis putus-putus kotak sebagai **batasan**, yang menunjukan kumpulan unit atau blok yang harus dilakukan *looping* atau perulangan sehingga didapat nilai pada umpan balik berdasarkan perbedaan antara aliran lima (5) dan enam (6) yang **mendekati** sama. Perulangan atau *looping* akan berhenti kalau persyaratan yang ditentukan yaitu nilai aliran lima dan enam mendekati sama misalnya kurang dari 10% perbedaannya. Jadi apabila ada sistem yang mengandung umpan balik (*recycle*) perlu dibuat batasan, blok-blok yang harus dilakukan proses perulangan.

**Permasalahan** selanjutnya, selain nilai aliran lima (5) dan enam (6) nilainya hampir mendekati sama, kapan perulangan dapat dianggap selesai dan benar menurut azas teknik kimia. Untuk menyelesaikan permasalahan ini, maka perlu referensi

dasar dalam semua perhitungan khususnya teknik kimia adalah neraca massa, artinya massa sebelum dan setelah sistem harus sama, dengan asumsi pada bagian ini tidak ada akumulasi. Oleh karenanya, berdasarkan pada gambar 1.17, karena sistem berada didalam garis putus-putus, maka jumlah massa pada aliran satu (1) harus **mendekati** sama dengan aliran empat (4), sehingga pada gambar 1.17 terlihat untuk aliran satu dan empat terlihat lebih tebal.




Gambar 1.17. PFD sistem dengan adanya umpan balik (*recycle*)

### 3.5.1. Aplikasi program sistem dengan blok dengan *Recycle*

Pada sub-bab ini akan dibahas mengenai kode program scilab untuk simulasi dan verifikasi hasil program apakah sudah benar berdasarkan alur (logika) algoritma program dan sudah benar berdasarkan pendekatan asas Teknik Kimia. Kedua verifikasi ini penting, karena secara logika program benar, akan tetapi secara pendekatan asas Teknik Kimia belum tentu benar. Berikut program1.19 yang merupakan kode program scilab. Selanjutnya akan dibahas kode program per bagian.

Pada program 1.19 terbagi menjadi 10 bagian. Pada bagian 1 merupakan eksekusi dari semua blok-blok (bentuk *.sci) dari sistem yang akan digunakan. Apabila semua blok tersebut

digunakan, maka tidak perlu dieksekusi satu-satu, tetapi dieksekusi dalam satu folder dengan **getd**, oleh karenanya semua blok sebaiknya ditempatkan dalam satu folder.

Selanjutnya, pada bagian 2, pada prinsip penyusunan Process Flow Diagram dari sistem yang akan digunakan atau disimulasikan. Jadi pada bagian tersebut terlihat variabel berupa JuKom (Jumlah Komponen), JuAru (Jumlah Arus) dan JUnOp (Jumlah alat – Unit Operation).

**Program1.19 : Aplikasi sistem terdiri dari tiga blok Unit dan dilengkapi Umpan Balik.**

| | |
|---|---|
| 1 | *//SisBlok_utama6.sce*<br><br>clear<br>clc<br>exec('D:/simsci/reaksto.sci');<br>exec('D:/simsci/mixer.sci');<br>exec('D:/simsci/spliter.sci');<br>exec('D:/simsci/tear1.sci');<br>exec('D:/simsci/Mol2Mass.sci');<br>exec('D:/simsci/Mass2Mol.sci');<br><br>*//bisa diganti dengan* **getd('D:/simsci/')**<br>*//agar semua *sci dapat di eksekusi* |
| 2 | *//CH4 O2 C2H6 CO2 H2O Daftar komponen*<br>*//Membuat PFD1*<br>JuKom=5;<br>*//Jumlah komponen*<br>JuAru=6;<br>*//Jumlah Arus - Aliran*<br>JUnOp=4;<br>*//Jumlah Unit Operasi - Alat*<br>BMK=[16 32 30 44 18] |

```
   //daftar BM Komponen

   //Kolom Flowrate: F
   FT=zeros(JuAru,1);
   XT=zeros(JuAru,JuKom);
3  rcl=1        //faktor hitung recycle

   //Kondisi Umpan:

   NoAru=1;
   FT(NoAru,1)=80
   //satuan mol/jam
   XT(NoAru,:)=[0.125 0.75 0.125 0 0]
   //persen mol

   [FTs,XTs,Ks]=Mol2Mass(FT,XT,BMK)
   //merubah mol ke mass
   FT=FTs
   XT=XTs
4  NoAruMa=[1,6; 2,0; 3,0; 5,0];
         //Baris=Posisi (nomor) alat
         //Kolom=Nomor arus masuk
   NoAruKe=[2,0; 3,0; 4,5; 6,0];
         //Baris=Posisi (nomor) alat
         //Kolom=Nomor arus Keluar

   ErFTear=100              //Error Tear dalam
   % - sebagai inisiasi
```

Pada bagian 3, menginisiasi jumlah recycle awal dengan nama variabel **rcl**. Dilanjutkan dengan memasukan nilai Aliran Umpan Segar (Feed), sebagaimana ilustrasi contoh diatas. Yang perlu mendapat perhatian, satuan dari Umpan segar dalam unit mole dan persen mole. Agar sesuai dengan prinsip Neraca Massa, maka satuan unit perlu diubah menjadi satuan berat, dengan

menggunakan blok **Mol2Mass**, sehingga yang masuk pada aliran mempunyai satuan massa bukan satuan mol.

Tahap 5 (lima) merupakan program perulangan (*looping*) yang dimulai dari tahap 5 sampai dengan tahap 10 untuk proses *recycle* umpan balik. Jadi pada tahap 10 merupakan akhir dari *loop* untuk bahasa pemograman pada *recycle* untuk bahasa proses teknik kimia, ditambah dengan variabel **rcl** sebagai faktor penghitung (***counter***).

**Program1.19 (lanjutan): Aplikasi sistem terdiri dari tiga blok Unit dan dilengkapi Umpan Balik.**

| | |
|---|---|
| 5 | `while ErFTear>0.01 then` |
| 6 | `disp('**********...**********')`<br>`//Alat ke 1:`<br>`UnOp(1)='Mixer';`<br>`//Bagian Masukkan:`<br>`if rcl==1 then`<br>`ukAruMa=size(NoAruMa(1,:));`<br>`JuAruMa=ukAruMa(1,2);`<br>`for i=1:JuAruMa`<br>`FMin(i,1)=FT(NoAruMa(1,i),1);`<br>`XMin(i,:)=XT(NoAruMa(1,i),:);`<br>`end`<br>`end`<br>`if rcl<>1 then`<br>`ukAruMa=size(NoAruMa(1,:));`<br>`JuAruMa=ukAruMa(1,2);`<br>`FMin`<br>`for i=1:JuAruMa`<br>`FMin(i,1)=FT(NoAruMa(1,i),1);`<br>`XMin(i,:)=XT(NoAruMa(1,i),:);`<br>`end`<br>`end` |

```
//disp('Massa in Mixer')
//------
[FHMix,XHMix]=mixer(FMin,XMin);
//utama mixer
//disp('Mixer')
    FT(NoAruKe(1,1),1)=FHMix;
//bentuk Total massa
    XT(NoAruKe(1,1),:)=XHMix;
//bentuk Komponen massa
disp('Mass from Mixer')
disp(FT)
disp(XT)
```

7

```
//karena masuk reaktor --> merubah massa ke
mol
    [FT,XT,KT]=Mass2Mol(FT,XT,BMK)
disp("Mol in reaktor")
//-----------------------------------------
----------------------
//Alat ke 2:
UnOp(2)='Reaktor';
//Bagian masukan reaktor:
NoAReMa=NoAruMa(2,1);
NoAReKe=NoAruKe(2,1);
Koef=[-1 -2 0 1 2; 0 -7/2  -1  2  3];
//Koefisien stokhiometri
Konv=[0.6; 0.8];
LC=[1 3];
FRin=FT(NoAReMa,1);
XRin=XT(NoAReMa,:);
   [Fout,Cout,
Bout]=reaksto(FRin,XRin,Konv,Koef,LC);
   FT(NoAReKe,1)=Fout;              //satuan
mol
   XT(NoAReKe,:)=Cout';             //satuan
mol
[FTs,XTs,Ks]=Mol2Mass(FT,XT,BMK)
//merubah mol ke mass
```

```
//disp(FTs)
    FT=FTs
    XT=XTs
//disp("Mass out Reaktor")
//disp(FT)
//disp(XT)
```

**Program1.19 (lanjutan): Aplikasi sistem terdiri dari tiga blok Unit dan dilengkapi Umpan Balik.**

```
8  //----------------------------------------
   ----------------------
   //Alat ke 3
   //Masukan untuk spliter
   Rs=0.7;            //Perbandingan yang
   keluar (4) dan recycle (5)
   NAMSp=NoAruMa(3,1);
   NAKSp=NoAruKe(3,:);
   FSpin=FT(NAMSp,1);
   XSpin=XT(NAMSp,:);
   //------
      [FSpO,XSpO]=spliter(FSpin,XSpin,Rs);
   for i=1:2
      FT(NAKSp(1,i),1)=FSpO(i,1);
      XT(NAKSp(1,i),:)=XSpO(i,:);
   end
   //disp('Splitter - massa')
   //disp(FT)
   //disp(XT)

9
   //----------------------------------------
   ----------------------
   //Alat ke 4

   NAMT(1)=NoAruMa(4,1);
   NAMT(2)=NoAruKe(4,1);
```

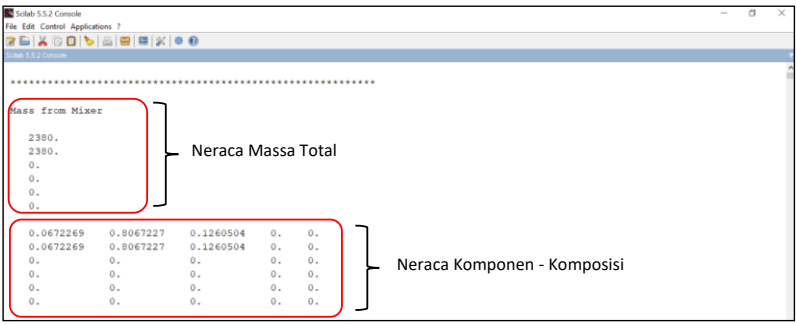
```
    NAKT=NoAruKe(4,1);

    for i=1:2
       FiT(i,1)=FT(NAMT(i),1);
       XiT(i,:)=XT(NAMT(i),:);
    end
    //disp("In Tear")
    //disp(FiT)
    //disp(XiT)

    [FOT,XOT]=tear1(FiT,XiT)
       FT(NAKT,1)=FOT
       XT(NAKT,:)=XOT

    disp('Tear - massa')
    disp("-------------------------------------
    ----------------")
    disp(rcl, "recycle ke= ")
    disp(FT)
    disp(XT)
    //menghitung tingkat perbedaan pada Tear
    ErFTear=(abs(FT(5,1)-FT(6,1)))*100/FT(5,1)
    disp('Error dlm %')
    disp(ErFTear)
    disp(rcl)
10  rcl=rcl+1
    end
```

Tahap 6 pada program 1.19 merupakan proses perhitungan pada blok **Mixer**, pada blok tersebut, diatur terlebih dahulu tentang matrik FT yaitu neraca massa total dari semua arus (aliran) dan XT merupakan neraca massa komponen dalam bentuk komposisi massa. Dilanjutkan dengan tahap 7 yang merupakan perhitungan untuk **Reaktor Konversi**, dimana massa yang akan masuk ke unit Reaktor diubah dulu satuannya dari

massa ke satuan mol dengan menggunakan blok **Mass2Mol**. Hasil keluaran dari Rektor kemudian diubah kembali menjadi satuan massa dengan menggunakan blok **Mol2Mass** sehingga pada saat akan masuk ke unit selanjutnya sudah dalam satuan massa.

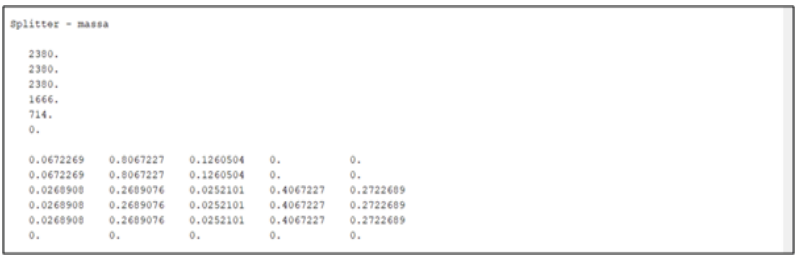
Tahap 8 pada program 1.19, merupakan perhitungan untuk unit **Splitter** dengan rasio perbandingan (**Rs**) antara yang dikeluarkan produk akhir (aliran 4) sebesar 0.7, sehingga untuk aliran 5 sebagai umpan balik (*recycle*) sebesar 0.3. Tahap 9 merupakan unit Virtual untuk menghitung neraca massa dan neraca komponen yang akan dikembalikan ke reaktor yang akan dicampurkan pada mixer. Dan tahap 10 merupakan looping yang akan kembali ke tahap 5 sampai kondisi aliran 5 dan 6 mendekati sama. Pada perhitungan ini digunakan tingkat kesalahan FT di aliran 5 dan 6 sekitar 0.01% yang dinyatakan dengan variabel **ErFTear**.

Hasil simulasi dari program 1.19 dapat dilihat pada gambar 1.18. Pada gambar 1.18 terlihat neraca massa dari Mixer yang hanya terdiri dari aliran 1 dan 2 (baris menunjukan urutan aliran), sedangkan pada aliran 3, 4, 5 dan 6 masih nol. Hal ini menunjukkan bahwa sistem perhitungan ini secara berurutan (sequence), yaitu menghitung Alat Unit 1 dahulu yaitu Mixer. Pada matrik berikutnya pada gambar 1.18 merupakan neraca komponen tiap aliran dimana pada kolom merupakan komponen dengan urutan: `//CH4 O2 C2H6 CO2 H2O`. Sebelum masuk pada Reaktor terlihat bahwa komponen $CO_2$ dan $H_2O$ masih nol pada aliran 1 dan 2.

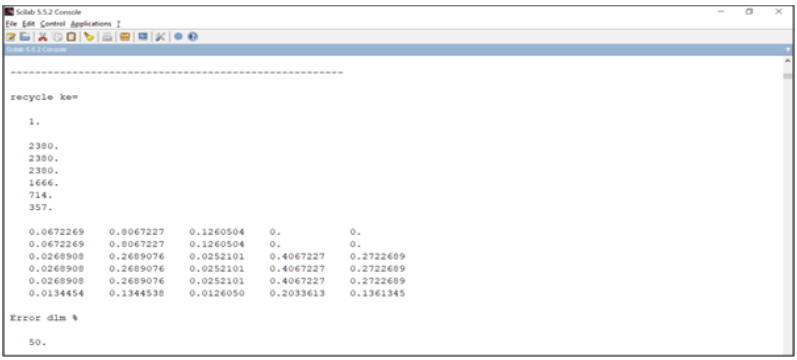



Gambar 1.18. Hasil simulasi sebelum looping

Pada gambar 1.19 merupakan hasil simulasi keluar dari Splitter alat ke 3 yang telah melewati Reaktor, akan tetapi belum terjadi *looping*. Terlihat pada neraca massa total maupun neraca massa komponen pada aliran ke 6 (keluar Tear, masuk Mixer) masih nol.

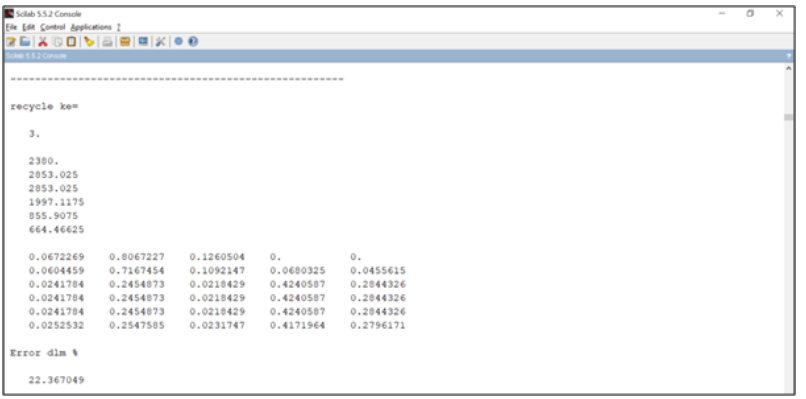



Gambar 1.19. Hasil simulasi setelah keluar dari Splitter.

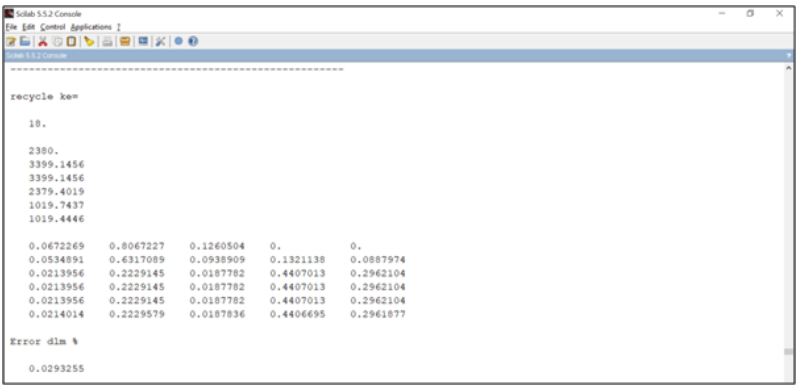



Gambar 1.20. Hasil simulasi setelah satu (1) kali loop.

Pada gambar 1.20 merupakan hasil simulasi setelah satu (1) kali putaran (loop) perhitungan komputasi. Pada metoda **Tear** digunakan metoda bagi dua (*bisection*) maka terlihat pada neraca massa aliran 5 sebesar 714 dan neraca massa pada aliran ke 6 sebesar 357, yang nilainya setengah dari aliran ke 5. Hal yang sama untuk neraca massa komposisi komponen. Dimana pada gambar tersebut tingkat kesalahan (*error*) sebesar 50% yang dihitung berdasarkan selisih aliran 5 dan 6 dibagi aliran 5 dalam persen. Pada gambar 1.21 merupakan hasil simulasi setelah tiga (3) kali putaran (looping), pada gambar terlihat tingkat kesalahan (*error*) sebesar 22.4%. Dan pada gambar 1.22 terlihat pada delapan belas (18) kali putaran (looping) dengan tingkat kesalahan (*error*) sebesar 0.03%. Nilai ini sebenarnya sudah cukup kecil, akan tetapi penulis akan menghentikan looping setelah nilai kesalahan 0.01%.



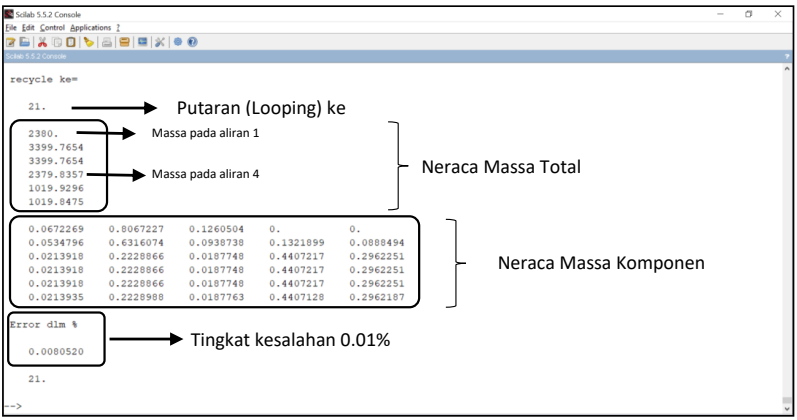

Gambar 1.21. Hasil simulasi setelah tiga (3) kali looping.




Gambar 1.22. Hasil simulasi setelah delapan belas (18) kali looping.

Hasil akhir dari simulasi dapat dilihat pada gambar 1.23, terlihat bahwa pada neraca massa total aliran 5 dan 6 hasilnya sudah mendekati yaitu 1019.9296 dan 1019.8475 dengan tingkat kesalahan atau selisih diantara kedua aliran sebesar 0.01%. Demikian juga halnya dengan neraca komponen komposisi. Untuk mendapat hasil tersebut diperlukan putaran (looping) sebanyak 21 kali putaran.

Sebagai validasi dari pada program secara ilmu dasar Teknik Kimia, bahwa dalam umpan balik (*recycle*), pada gambar 1.17 nilai massa yang masuk pada kotak putus-putus harus mendekati sama dengan nilai yang keluar dari kotak tersebut, atau dikatakan nilai aliran 1 dan aliran 4 mendekati sama. Pada gambar 1.23 terlihat nilai aliran 1 sebesar 2380 dan nilai aliran 4 sebesar 2379.8357. Nilai tersebut kalau dihitung tingkat kesalahan (perbedaannya) sebesar 0.0069%.




Gambar 1.23. Hasil akhir simulasi setelah duapuluh satu (21) kali looping.

Berdasarkan pada hasil simulasi program dengan menggunakan sistem blok yang terdapat bagian umpan balik (*recycle*) terlihat bahwa ada dua macam validasi yang perlu dilihat, pertama validasi kebenaran pada logika pemrograman hal in dengan membandingkan nilai aliran 5 dan 6 mendekati sama, selanjutnya validasi kedua kebenaran berdasarkan ilmu Teknik Kimia dengan neraca massa total yaitu dengan membandingkan nilai aliran 1 dan aliran 4 mendekati sama. Pada simulasi diatas validasi pemrograman tingkat perbedaaan sebesar 0.008%, sedangkan validasi keilmuan Teknik Kimia dengan tingkat perbedaan sebesar 0.0069%.

## 4. Kesimpulan

Pada artikel ini telah dituliskan mengenai beberapa macam kasus dalam menyusun lima macam model dan simulasinya. Dimana dari keempat model pertama adalah penyusunan blok atau bagian dari sistem. Dimulai dari kasus reaktor bioproses, sebagai bagian dari blok atau unit. Pada kasus ini dibahas mengenai penentuan parameter berdasarkan pada

hasil ekperimen lapangan atau laboratorium dengan menggunakan subprogram optimasi tingkat kesalahan antara data ekperimen dan data perhitungan. Sedangkan pada model kelima (5) merupakan aplikasi dari penggabungan tiga macam unit blok dan satu unit virtual yang merupakan bagian proses perhitungan untuk sistem yang dilengkapi dengan umpan balik (recycle). Untuk kode program blok ditandai dengan *.sci dapat dilihat pada bagian ini setelah referensi.

## 5. Referensi


Alt, R. and Markov, S. (2012) 'Theoretical and computational studies of some bioreactor models', *Computer and Mathematics with Applications*, 64, pp. 350–360. doi: 10.1016/j.camwa.2012.02.046.

Bequette, B. W. (1998) *Process Dynamics: Modeling, Analysis, and Simulation*. Prentice Hall PTR.

Constantinides, Alkis., Mostoufi, N. (1999) *Numerical Methods for Chemical Engineers with MatLab Applications*. Prentice Hall PTR.

Esfahanian, M. *et al.* (2016) 'Mathematical modeling of continuous ethanol fermentation in a membrane bioreactor by pervaporation compared to conventional system : Genetic algorithm', *Bioresource Technology*, 212, pp. 62–71.

Fogler, H. S. (2006) *Elements of Chemical Reaction Engineering*. Prentice Hall PTR.

phoudha (2020) *Data fitting leastsq vs lsqrsolve, don't success*, *Fossee Forums*. Available at: https://forums.fossee.in/question/1349/.

Sasongko, S. (2010) *Metode Numerik dengan Scilab*. Andi Offset, Yogyakarta.

**Program1.20 : Blok mixer.**

D:/simsci/mixer.sci

```
function [FOmix,XOmix]=mixer(Fin,Xin);
//Modul pencampur
   FinT=Fin';
   FOmix=sum(FinT);
   XOmix=FinT*Xin/FOmix;
endfunction
```

**Program1.21 : Blok Rektor Konversi.**

D:/simsci/reaksto.sci

```
function [Fmout, XCout,
Bout]=reaksto(Fmin,XCin,Konv,Koef,LR);
   //Reaktan Stokhiometri
   //Fm = molar flowrate total = mol/waktu
   //B = molar flowrate komponen = mol/waktu
   //XC = fraksi mol komponen
   //LR: Limiting reaktan, dalam bentuk baris,
misal: LR=[1 3]
   //Konv dalam bentuk matrix kolom

   Bin=Fmin.*XCin;
   ukr=size(LR);
   jr=ukr(1,2);
   for i=1:jr
      BLR(1,i)=Bin(1,LR(1,i));
   end
   BLR=BLR';
   Xblr=Konv.*BLR;

   Koef=Koef';    //ubah ke bentuk kolom
   Brx=Koef*Xblr;    //flowrate reaksi
   Bout=Bin'+Brx;
   Fmout=sum(Bout);
   XCout=Bout./Fmout;
endfunction
```

**Program1.22 : Blok Splitter.**

D:/simsci/splitter.sci

```
function [FSpO,XSpO]=spliter(FSpin,XSpin,Rs);
   FSpO(1,1)=FSpin*Rs;
   FSpO(2,1)=FSpin-FSpO(1,1);
   XSpO(1,:)=XSpin;
   XSpO(2,:)=XSpin;
endfunction
```

**Program1.23 : Blok Tear (recycle).**

D:/simsci/tear1.sci

```
function [FOT, XOT, rF]=tear1(FiT, XiT)
   FOT=sum(FiT)./2;
   XOT=(sum(XiT,'r'))./2;

   rF=(abs(FOT(1,1)-FiT(2,1)))/FOT(1.1)*100
endfunction
```

**Program1.24 : Blok Mol2Mass (konversi satuan dari mol ke massa).**

D:/simsci/Mol2Mass.sci

```
function [FTs, XTs, KTs]=Mol2Mass(FT, XT, BM)
//BM bentuk vektor baris
//Baris --> Nomor arus
//Kolom --> Komponen
//XT, Xts --> bentuk matrik --> bentuk
komposisi
//FT, Fts --> bentuk vektor kolom --> bentuk
total mol atau massa
Ukran=size(XT);
JuAru=Ukran(1,1);
JuKom=Ukran(1,2);

for NA=1:JuAru
```

```
    KKT(NA,:)=FT(NA,1).*XT(NA,:)
//Komponen mol
    KTs(NA,:)=KKT(NA,:).*BM
//Komponen massa
    FTs(NA,:)=sum(KTs(NA,:))
    if FTs(NA,:) <> 0 then
CFTs(NA,:)=1/FTs(NA,:);
        else CFTs(NA,:)=FTs(NA,:);
    end
    XTs(NA,:)=KTs(NA,:)*CFTs(NA,:)
end

endfunction
```

**Program1.25 : Blok Mass2Mol (konversi satuan dari massa ke mol).**

D:/simsci/Mass2Mol.sci

```
function [FTh, XTh, KTh]=Mass2Mol(FTs, XTs, BM)
//BM bentuk vektor baris
//Baris --> Nomor arus
//Kolom --> Komponen
//XT, Xts --> bentuk matrik --> bentuk
komposisi
//FT, Fts --> bentuk vektor kolom --> bentuk
total mol atau massa
Ukran=size(XTs);
JuAru=Ukran(1,1);
JuKom=Ukran(1,2);
BMKs=1 ./BM;
for NA=1:JuAru
    KKTs(NA,:)=FTs(NA,1).*XTs(NA,:)
//Komponen massa
    KTh(NA,:)=KKTs(NA,:).*BMKs
//Komponen mol
    FTh(NA,:)=sum(KTh(NA,:))
```

```
    if FTh(NA,:) <> 0 then
CFTh(NA,:)=1/FTh(NA,:);
        else CFTh(NA,:)=FTh(NA,:);
    end
    XTh(NA,:)=KTh(NA,:)*CFTh(NA,:)
end
endfunction
```

# BAB II

# Integrasi *property packages* CAPE-OPEN simulator dengan GUI-Scilab

Setia Budi Sasongko (sbudisas@live.undip.ac.id)

Kata kunci (*Keywords*):
Scilab, CAPE-OPEN, GUI Scilab – APG (Antarmuka Pengguna Grafis).

Abstraksi

Permasalahan pemodelan sistem proses industri kimia adalah mendapatkan data properti kimia, fisika, kesetimbangan fase. Data properti paket tersebut pada umumnya telah disediakan pada perangkat lunak simulasi. Pada artikel ini, tujuannya adalah melakukan integrasi properti paket dari perangkat lunak simulator CAPE-OPEN, DWSIM pada Scilab 5.5.2. Integrasi lebih bersifat *user-friendly* dengan menggunakan bahasa pemrograman GUI (*Graphical User Interface*) atau Antarmuka Pengguna Grafis (APG) dari Scilab.

## 1. Pendahuluan.

CAPE-OPEN merupakan perangkat lunak simulasi untuk bidang Teknik Kimia yang merupakan kepanjangan dari ***Computer Aided Process Engineering.*** CAPE-OPEN salah satu perangkat lunak simulasi yang tidak berbayar, hal ini sangat berguna khususnya untuk pendidikan, penelitian maupun pengembangan perangkat lunak dalam skala industri. Perangkat lunak simulasi digunakan untuk perancangan proses, untuk simulasi dari

operator proses pabrik, dapat juga digunakan sistem optimasi sistem proses (Baten, Van, Taylor and Kooijman, 2010). Perangkat lunak sistem simulasi pada umumnya sudah dilengkapi dengan ***Property Packages*** yang berisi data komponen-komponen dengan karakteristik kimia maupun fisika, karakteristik kesetimbangan Uap-Cair berdasarkan pada hubungan termodinamika dari campuran komponen pada kondisi ideal maupun non-ideal. Perangkat lunak sistem simulasi juga dilengkapi dengan aplikasi perhitungan secara umum untuk ***Unit Operation***.

Selain perangkat lunak simulasi, dalam bidang Teknik (Sasongko and Luqman, 2018)biasanya menggunakan **perangkat lunak aplikasi perhitungan numerik** seperti Matlab™, Scilab, NumPy, GNU Octave dan lain sebagainya. Perangkat lunak seperti Matlab™, Scilab selain digunakan untuk analisis numerik, digunakan juga untuk visualisasi data, pengembangan algoritma dan pemodelan (Hugo Magalhaes da Fonseca, 2019). Scilab perangkat lunak komputasi numerik yang bersifat bebas berbayar, merupakan salah satu bahasa pemograman level-tinggi seperti Matlab™ dalam menyelesaikan berbagai macam permasalahan model matematika.

### 1.1. Permasalahan dan tujuan

Penggunaan perangkat lunak simulasi komersial seperti Hysys, ChemCAD, Aspen pada umumnya sangat *user-friendly*, akan tetapi persamaan model pada umumnya bersifat ***black-box,*** artinya persamaan model tidak dapat diakses oleh pengguna (Baten, Van, Taylor and Kooijman, 2010). Selain itu perangkat lunak simulasi bersifat **general (umum)**, sehingga untuk pengembangan penelitian maupun untuk proses pembelajaran menjadi sangat terbatas. Sedangkan perangkat lunak Scilab penyelesaian model matematis atau aplikasi numerik untuk

pengembangan pada bidang teknik kimia sangat terbatas atau bahkan tidak tersedia data properti paket komponen, keseimbangan fase termodinamika. Properti paket ini pada umumnya tersedia pada perangkat lunak simulasi, bahkan untuk perangkat lunak simulasi komersial data properti paket sangat lengkap.

Oleh karenanya, tujuan utama pada artikel ini adalah integrasi antara **perangkat lunak simulasi** yang memiliki properti paket pada **perangkat lunak numerik** model matematika, yang menggunakan perangkat lunak bebas berbayar dan open source ***free*** dan ***open source software*** **(FOSS).** Kebaharuan dari sistem ini selain penggabungan antara perangkat lunak CAPE-OPEN dengan Scilab untuk memanfaatkan properti paketnya adalah dikembangkan dengan menggunakan **GUI** (***Graphical User Interface***) sehingga akan lebih memudahkan pengguna.

## 2. Material dan Metoda

Artikel ini disusun berdasarkan pengalaman (*experience*) dalam mengembangkan program komputer, sehingga tingkat keberhasilannya tinggi (karena harus ada hasil yang sesuai). Peralatan yang digunakan untuk menyusun artikel ini adalah Laptop merk Asus dengan spesifikasi:

- Processor: Intel® Core™ i7-7500U CPU @2.70GHz 2.90 GHz
- Installed RAM 16.0 GB
- System type 64-bit Operating System, x64-based processor.

Perangkat lunak yang digunakan adalah:

- Windows spesifikasi:
  - Edition: Windows 10 Home Single Language
  - Version: 1903

- Perangkat lunak numerik – modeling (Free and Open Source Software):
  - Scilab 5.5.2 versi Window 64 bits
- Perangkat lunak simulasi (Free and Open Source Software)
  - COFE version 3.4.0.0 (x64) – CAPE-OPEN Flowsheeting environment
  - TEA property package – Thermodynamic for Engineering Application
  - ChemSep 8.23 LITE – Modeling Separation Process
  - DWSIM Open Source Process Simulator, Version 6.1 Update 9, Copyright (c) 2017-2020 Daniel Medeiros.

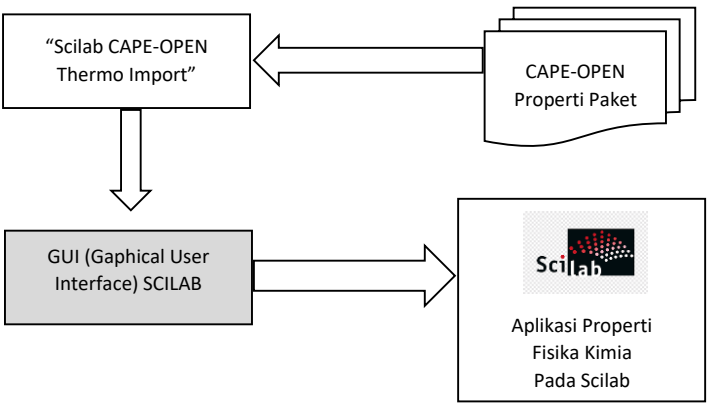
COCO (CAPE-OPEN to CAPE-OPEN) merupakan kumpulan perangkatan lunak simulasi steady state proses kimia yang bebas bayar, yang terdiri dari empat komponen:

1. COFE untuk flowsheeting engine
2. TEA (Thermodynamics for Engineering Applications) untuk system Thermodinamika
3. COUSCOUS koleksi unit operation (peralatan bidang Teknik Kimia)
4. CORN untuk paket reaksi numerik

Distribusi dari COCO juga selalu menyertakan perangkat lunak ChemSep LITE. Dimana ChemSep merupakan perangkat lunak proses separasi destilasi berdasarkan pada sistem kesetimbangan. Untuk mendapatkan perangkat lunak ini dapat diunduh pada alamat: http://www.chemsep.com/ atau https://www.cocosimulator.org/ (*CAPE-OPEN*, 2020).

Metode yang digunakan pada artikel ini adalah mengintegrasikan paket properti dari perangkat lunak simulasi seperti CAPE-OPEN kedalam scilab. Apabila pada komputer

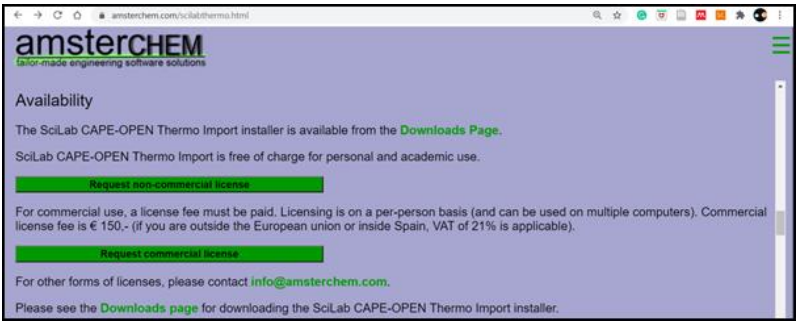
perangkat simulasi lebih dari maka perlu dipilih oleh pengguna, dalam hal ini digunakan metoda GUI, Graphical User Interface. Secara garis besar metoda yang digunakan dapat dilihat pada gambar 2.1.




Gambar 2.1: Metode Integrasi CAPE-OPEN dan Scilab dengan GUI

Aplikasi "Scilab CAPE OPEN thermo Import" dapat digunakan untuk mengimport CAPE-OPEN versi 1.1 dan properti paket fisikal dan kimia pada Scilab. Untuk mendapatkan perangkat lunak tersebut dapat diunduh pada alamat: https://www.amsterchem.com/downloads.html, sebagaimana ditunjukkan pada gambar 2.2. Perangkat lunak tersebut dapat digunakan selama 30 hari, agar dapat digunakan tanpa batas waktu, maka dapat menggunakan lisensi non-komersial untuk personal atau penguna akademi, sebagaimana ditunjukan pada gambar 2.3.


Gambar 2.2. File Scilab CAPE-OPEN dari www.amsterchem.com

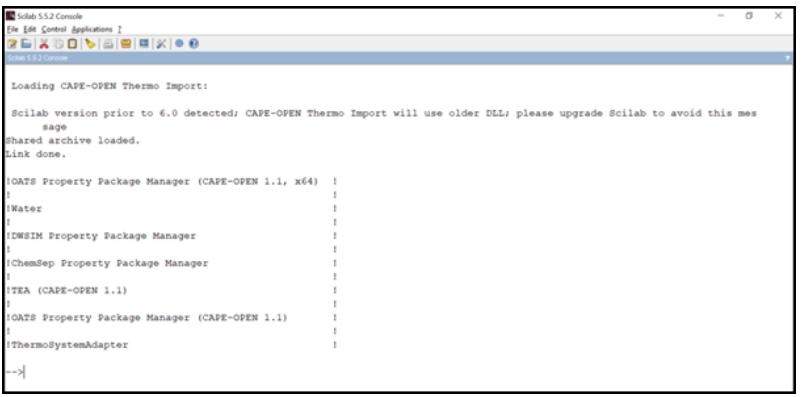



Gambar 2.3. Lisensi Scilab CAPE-OPEN Thermo Import dari www.amsterchem.com/scilabthermo.html.

Setelah file "Scilab ThermoImport.2.0.0.9.exe" dieksekusi, maka akan terbentuk file: "capeOpenThermoImportInit.sci" yang perlu dieksekusi pada Scilab, sebagaimana Program1. Pada program ***loading*** CAPE-OPEN Thermo Import dengan perintah exec, kemudian dilanjutkan dengan membuka paket yang ada pada sistem di komputer (tiap komputer bisa berbeda hasil, tergantung aplikasi CAPE-OPEN yang ada pada sistem) dengan perintah: capeOpenPackageManagers(), setelah itu hasilnya ditampilkan dengan perintah: disp, hasil eksekusi dapat dilihat pada gambar 2.4. Pada gambar tersebut terlihat, pada sistem komputer tersedia tujuh (7) program yang dapat terintegrasi dengan CAPE-OPEN dan Scilab.

Program2.1: Eksekusi file: “capeOpenThermoImportInit.sci”

```
clc, clear
exec('C:\Program Files\Scilab CAPE-OPEN Thermo
Import\capeOpenThermoImportInit.sci');
fileCapeku=capeOpenPackageManagers();
disp(fileCapeku)
```




Gambar 2.4: Integrasi CAPE-OPEN Thermo Import pada Scilab tanpa GUI

Untuk memilih program “property package” yang akan digunakan pada file manager, maka digunakan perintah input, sebagaimana yang ditunjukkan pada program 2.2.

Program2.2: Pemilihan program “property package” dengan input.

```
clc, clear
exec('C:\Program Files\Scilab CAPE-OPEN Thermo
Import\capeOpenThermoImportInit.sci');
fileCapeku=capeOpenPackageManagers();
disp(fileCapeku)

mprintf('\n')
```

```
plh=input('Pilih property package= ')
propack=fileCapeku(plh)
disp(propack)
```

Agar penguna (*user*) dapat lebih interaktif dengan program, maka digunakan GUI (*Graphical User Interface*) pada Scilab, yaitu **x_choose**, sebagaimana ditunjukkan pada program 2.3. Dengan menggunakan perintah tersebut, maka akan tampil window dan pengguna hanya meng-klik dua kali bagian yang dipilih, dan akan disimpan pada variabel: pilSoPa, misalkan **TEA (CAPE-OPEN 1.1)**. Selanjutnya akan dipilih file yang berisi Component pilihan dan konstanta pada property packed yang dipilih. Perintah **xchoose** digunakan untuk memilih variable yang tersedia, sedangkan untuk mengisi nilai variabel dapat digunakan perintah **x_mdialog**, permasalahan angka yang dimasukan dibaca sebagai **string** bukan numerik, oleh karenanya perlu diubah dengan perintah **evstr**.

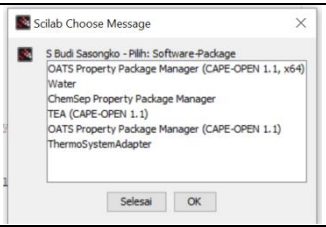
Program2.3: Aplikasi perintah **x_choose** pada GUI-Scilab.

```
clc, clear
exec('C:\Program Files\Scilab CAPE-OPEN Thermo
Import\capeOpenThermoImportInit.sci');

managCapeku=capeOpenPackageManagers();       //Daftar
software yang support CAPE-OPEN dan manager pada sistem
mprintf('\n')

judul1='S Budi Sasongko - Pilih: Software-Package'
pilSoPa=x_choose(managCapeku,judul1,'Selesai')  //memilih
Software paket dengan GUI
SoPa=managCapeku(pilSoPa)                    //menyimpan
```

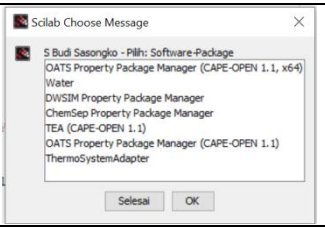
```
software paket

ppSBS1=capeOpenPackages(SoPa)              //membuka
Softaware Package

judul2='S Budi Sasongko - Pilih: Property Package'
pilPP=x_choose(ppSBS1,judul2,"selesai")      //memilih properti
paket pada software pilihan

hslPP=capeOpenGetPackage(SoPa,ppSBS1(pilPP))   //membuka
paket

//menampilkan komponen pilihan:
disp(capeOpenCompounds(hslPP))

//menampilkan konstanta pada PP
list1=capeOpenConstantList(hslPP)
```

## 3. Hasil dan Diskusi

Hasil dari program GUI Scilab integrasi program-program yang mendukung CAPE-OPEN sebagaimana dapat dilihat pada gambar 2.5. Pada gambar tersebut terlihat adanya perbedaan antara bagian (a) dan (b), belum ada program DWSIM Property Package Manager, hal ini disebabkan program tersebut belum teregistrasi dengan CAPE-OPEN, sehingga perlu adanya lisensi sebagaimana yang ditunjukkan pada gambar 2.3 diatas. Untuk memilih paket program yang diinginkan, klik dua kali pada program atau klik sekali kemudian tekan tombol OK, script dapat dilihat pada program2.4.




a. DWSIM belum muncul, krn tidak ada dalam system, atau belum registry




b. Ada DWSIM, sudah ada dalam sistem dan harus registry.

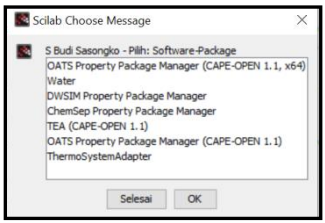
Gambar 2.5. Integrasi program yang mendukung CAPE-OPEN

Program2.4: Aplikasi perintah **x_choose** pada GUI-Scilab.

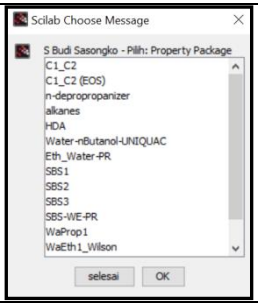
```
clc, clear
exec('C:\Program Files\Scilab CAPE-OPEN Thermo
Import\capeOpenThermoImportInit.sci');

managCapeku=capeOpenPackageManagers();        //Daftar
software yang support CAPE-OPEN
mprintf('\n')

judul1='S Budi Sasongko - Pilih: Software-Package'
pilSoPa=x_choose(managCapeku,judul1,'Selesai')  //memilih
Software paket dengan GUI
SoPa=managCapeku(pilSoPa)                  //menyimpan
software paket

ppSBS1=capeOpenPackages(SoPa)              //membuka
Softaware Package

judul2='S Budi Sasongko - Pilih: Property Package'
pilPP=x_choose(ppSBS1,judul2,"selesai")       //memilih properti
paket pada software pilihan
```

```
hslPP=capeOpenGetPackage(SoPa,ppSBS1(pilPP))   //membuka paket

//menampilkan komponen pilihan:
disp(capeOpenCompounds(hslPP))

list1=capeOpenConstantList(hslPP)
```

Salah satu program yang digunakan disini adalah TEA (CAPE OPEN 1.1) atau program *Thermodynamics for Engineering Applications*, seperti pada gambar 2.6. Pada program ini perlu dibuat komponen yang kita pilih, property packed yang digunakan; Tahapan pegunaan sebagaimana ditunjukkan pada gambar 2.7.

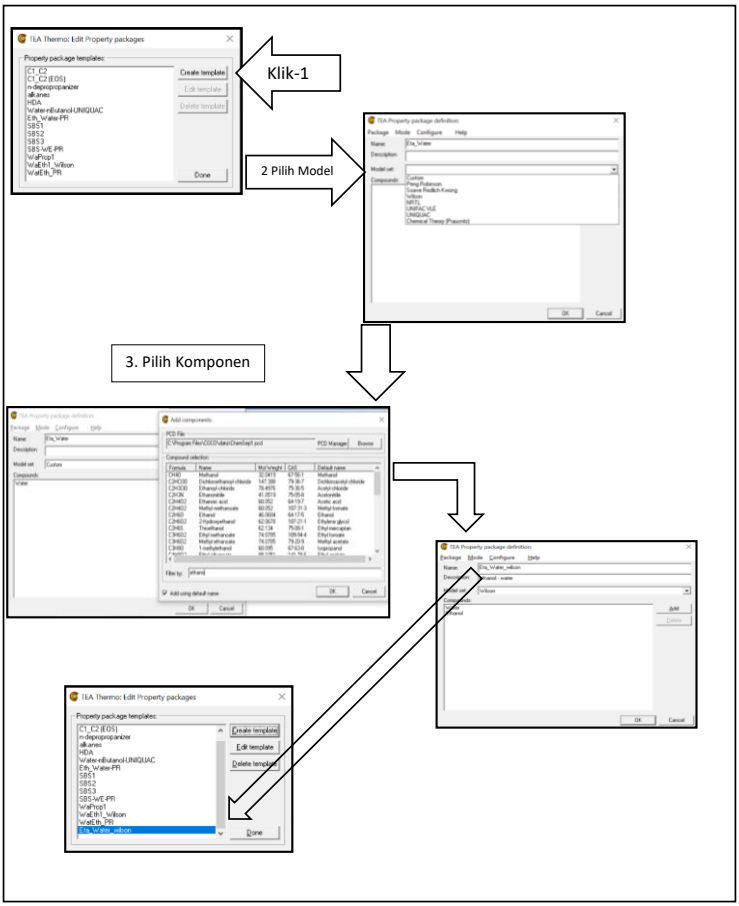



a. Penggunaan **x_choose** tahap 1, dipilih TEA (CAPE-OPEN 1.1)




b. Penggunaan **x_choose** tahap 2, dipilih Property Package yang tersedia

Gambar 2.6. Pemilihan TEA (CAPE-OPEN 1.1)



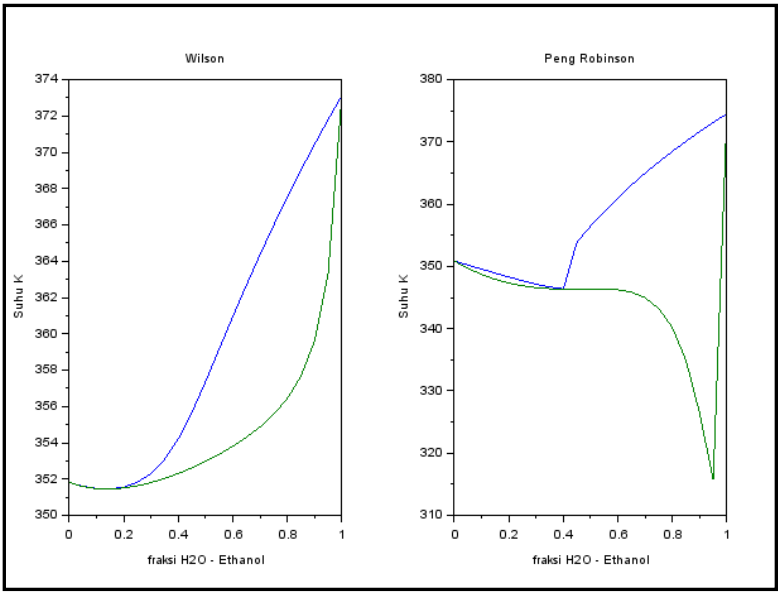

Gambar 2.7. Penambahan **Property package tempales**.

Setelah dapat mengambil paket properti, tahap selanjutnya adalah menggunakan properti tersebut untuk aplikasi simulasi kimia industri. Misalkan akan dilihat karakteristik campuran untuk dua (2) komponen azeotrop yaitu “ethanol dan water” dengan membuat grafik untuk properti fisika yang berbeda.

**Program2.5: Aplikasi kesetimbangan properti dari campuran "Water-Ethanol".**

```
clc, clear
//TEA CAPE-OPEN
exec('C:\Program Files\Scilab CAPE-OPEN Thermo
Import\capeOpenThermoImportInit.sci');

managCapeku=capeOpenPackageManagers();
//Daftar software yang support CAPE-OPEN dan manager
pada sistem
mprintf('\n')

//TEA - CAPEOPEN 1.1.
SoPa=managCapeku(5)              //menyimpan software
paket

ppSBS1=capeOpenPackages(SoPa)          //membuka
Softaware Package

hslPP=capeOpenGetPackage(SoPa,ppSBS1(13))
//membuka paket Wilson 13
//menampilkan komponen pilihan:
disp(capeOpenCompounds(hslPP))

frak=0:0.05:1;
X=[frak' 1-frak']
P=101325;

tbub=capeOpenEquilibriumProp(hslPP,"temperature",X,"p
ressure",P,"vaporFraction",0)
tdew=capeOpenEquilibriumProp(hslPP,"temperature",X,"p
```

```
ressure",P,"vaporFraction",1)

subplot(1,2,1)
plot(frak',[tdew tbub])
xtitle("Wilson","fraksi H2O - Ethanol","Suhu K")
hslPP=capeOpenGetPackage(SoPa,ppSBS1(14))
//membuka paket    PR  14
//menampilkan komponen pilihan:
disp(capeOpenCompounds(hslPP))

frak=0:0.05:1;
X=[frak' 1-frak']


P=101325;          //x_mdialog ... evstr

tbub=capeOpenEquilibriumProp(hslPP,"temperature",X,"p
ressure",P,"vaporFraction",0)
tdew=capeOpenEquilibriumProp(hslPP,"temperature",X,"p
ressure",P,"vaporFraction",1)

subplot(1,2,2)
plot(frak',[tdew tbub])
xtitle("Peng Robinson","fraksi H2O - Ethanol","Suhu K")
```

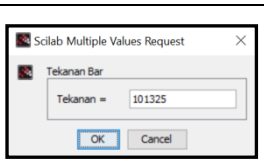



Gambar 2.8. Kesetimbangan “Water-Ethanol” antara Wilson dan Peng-Robinson.

Apabila akan membanding untuk tekanan yang berbeda dapat digunakan perintah “**x_mdialog**”, dimana nilai tekanan yang diinginkan dapat dimasukkan langsung oleh pengguna, kecuali pengguna tidak akan mengganti, maka akan ada niali default. Permasalahannya nilai angka yang dimasukkan dibaca sebagai string (bukan numerik atau angka), oleh karenanya perlu diubah menjadi numerik dengan perintah “**evstr**”.

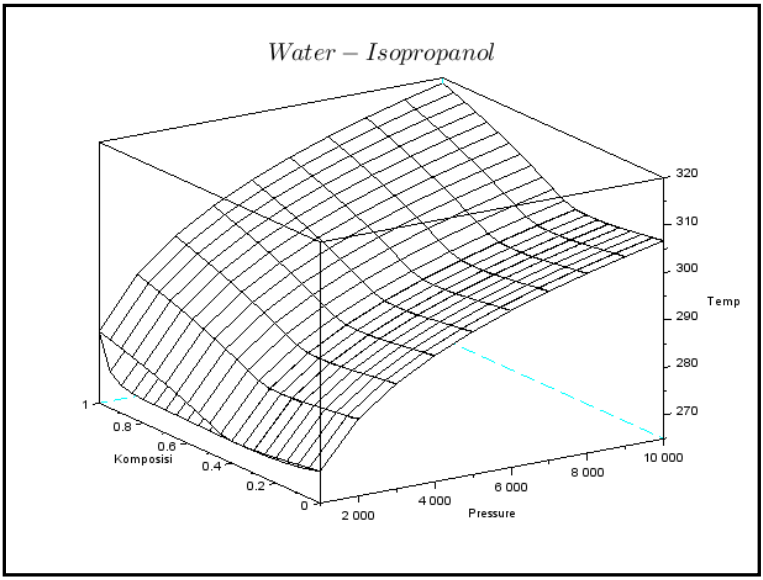
Program2.6: Aplikasi GUI Scilab dengan **x_mdialog** dan **evstr** serta tampilan GUI sebagaimana sebelah kanan.

```
jdlDia="Tekanan Bar"
P=x_mdialog(jdlDia,"Tekanan =
","101325")
P=evstr(P)
```



Program2.7: Visualisasi water Isopropanol dalam tiga dimensi.

```
clc, clear
exec('C:\Program Files\Scilab CAPE-OPEN Thermo
Import\capeOpenThermoImportInit.sci');
mgrs=capeOpenPackageManagers()
handle=capeOpenGetPackage("TEA (CAPE-OPEN
1.1)","WaProp1")

disp(capeOpenCompounds(handle)) //menampilkan senyawa
pada system

frac=(0:0.05:1)'
X=[frac 1-frac]
P=[1e3:1e3:1e4];
tbub=[];
tdew=[];

for p=P
 tbub=[tbub
capeOpenEquilibriumProp(handle,'temperature',X,'pressure',p,'vap
orFraction',0)]
 tdew= [tdew
capeOpenEquilibriumProp(handle,'temperature',X,'pressure',p,'vap
orFraction',1)]
end

mesh(P,frac,tbub);
mesh(P,frac,tdew);
xtitle("$\huge Water - Isopropanol
$","Pressure","Komposisi","Temp")
```




Gambar 2.9. Kesetimbangan “Water-Isopropanol”.

Program2.7 merupakan aplikasi penggunaan “**capeOpenEquilibriumProp**” untuk menghitung kesetimbangan dari komponen campuran pada fraksi uap = 0, yang disimpan dalam variabel **tbub** atau suhu pada titik bubble dan fraksi uap = 1, yang disimpan pada variabel **tdew** atau suhu pada titik dew yang dilakukan untuk berbagai tekanan, hasilnya dapat dilihat kurva kesetimbangan tiga dimensi dari komponen water-isopropanol.

## 4. Kesimpulan

Pada artikel ini telah didemonstrasikan integrasi antara CAPE OPEN COCO simulator dengan Scilab sebagai perangkat lunak modeling numerik.

- Penggunaan basis data pada property package dari perangkat lunak yang mendukung CAPE-OPEN dengan menggunakan “*cape open thermo import*” telah dikembangkan.

- Pengembangan dengan menggunakan GUI-Scilab, **x_choose**; **x_mdialog** pada artikel ini untuk mempermudah pengguna dalam berinteraksi dengan program.

## 5. Referensi


Baten, Van, J., Taylor, R. and Kooijman, H. (2010) 'Using Chemsep, COCO and modeling tools for versatility in custom process modeling', *AIChE annual meeting*. Available at: http://www.cocosimulator.org/downloads/SaltlakeCityAIChE2010ExtendedAbstract_Chemsep_COCO.pdf.

*CAPE-OPEN* (2020). Available at: http://www.cocosimulator.org/.

Hugo Magalhaes da Fonseca, A. de O. C. (2019) 'Development of a simulation module for the process of Alcoholic fermentation using a free software', *Enciclopedia Biosfera*, 16, pp. 2314–2326. doi: 10.18677/EnciBio.

Sasongko, S. and Luqman, B. (2018) 'Improving the Student Competency of the " Modeling and Computation Process " Course by Using Open Source', *MATEC Web of Conferences 156, 07003 (2018) RSCE 2017*, 7003, pp. 2017–2019.

# BAB III

# Pengembangan modul CAPE OPEN-COCO simulator dengan Scilab

Setia Budi Sasongko (sbudisas@live.undip.ac.id)

Kata kunci (*Keywords*):
Scilab, CAPE-OPEN.

Abstraksi

Perangkat lunak simulator khususnya untuk bidang teknik kimia pada umumnya dibuat secara general, sehingga untuk keperluan khusus harus dikembangkan. Untuk mengatasi hal tersebut, perlu adanya penggabungan antara perangkat lunak simulasi dengan perangkat lunak perhitungan numerik seperti Scilab. Tujuan utama dari artikel ini adalah menyusun program (*script*) untuk suatu modul atau ***Unit Operation*** dengan menggunakan program Scilab pada perangkat lunak simulasi CAPE-OPEN.

Hasil dari studi ini adalah penggabungan antara Scilab sebagai modul Unit Operation pada CAPE OPEN simulator dan dapat di simulasikan dengan beberapa kondisi yang diinginkan oleh pengguna.

## 1. Pendahuluan.

Perangkat lunak simulasi bidang Teknik Kimia yang bersifat bebas bayar adalah CAPE-OPEN (***Computer Aided Process Engineering***) digunakan untuk memahami, merancang, mengembangkan unit maupun sistem proses Teknik Kimia. Sehingga perangkat lunak ini dapat digunakan untuk mahasiswa sebagai bagian dari perancangan tugas akhir, peneliti untuk

mengembangkan, merancang maupun mengoptimasi sistem proses, dan profesional maupun praktisi dalam sistem proses (Baten, Van, Taylor and Kooijman, 2010).

Perangkat lunak simulasi pada umumnya memiliki fitur yang sangat menarik, basis data lengkap, mudah dalam pengoperasiannya dan masih banyak kelebihannya, apalagi yang berbayar. Kelebihan dari perangkat lunak simulasi ini bersifat *user friendly* karena menggunakan visualisasi grafis dengan pengguna atau APG – Antarmuka Pengguna Grafis.

### 1.1. Permasalahan dan tujuan

Salah satu kendala dari perangkat lunak sistem simulasi ini bersifat *black-box* untuk setiap unitnya artinya pengguna mengubah-ubah input maupun parameter unit dan akan mendapatkan hasil sebagai bagian dari output unit. Sistem unit bersifat general atau umum, hal ini akan menjadi masalah bagi mahasiswa karena kurang memahami filosofi unit secara mendalam, sehingga penggunaan dari alat simulator ini bersifat coba-coba (*trial-error*). Berdasarkan hal tersebut, pada artikel ini bertujuan mengembangkan modul unit operation dengan menggunakan bahasa Scilab pada perangkat lunak simulasi CAPE OPEN.

## 2. Material dan Metoda

Artikel ini disusun berdasarkan pengalaman (*experience*) dalam mengembangkan program komputer, sehingga tingkat keberhasilannya tinggi (karena harus ada hasil yang sesuai). Peralatan yang digunakan untuk menyusun artikel ini adalah Laptop dengan spesifikasi:

Laptop merk Asus dengan spesifikasi:

- Processor: Intel® Core™ i7-7500U CPU @2.70GHz 2.90 GHz
- Installed RAM 16.0 GB
- System type 64-bit Operating System, x64-based processor.

Perangkat lunak yang digunakan adalah:

- Windows spesifikasi:
  - Edition: Windows 10 Home Single Language
  - Version: 1903
- Perangkat lunak numerik – modeling (Free and Open Source Software):
  - Scilab 5.5.2 versi Window 64 bits
- Perangkat lunak simulasi (Free and Open Source Software)
  - COFE version 3.4.0.0 (x64) – CAPE-OPEN Flowsheeting environment
  - DWSIM Open Source Process Simulator, Version 6.1 Update 9, Copyright (c) 2017-2020 Daniel Medeiros.

COCO (CAPE-OPEN to CAPE-OPEN) merupakan kumpulan perangkatan lunak simulasi steady state proses kimia yang bebas bayar, yang terdiri dari empat komponen:

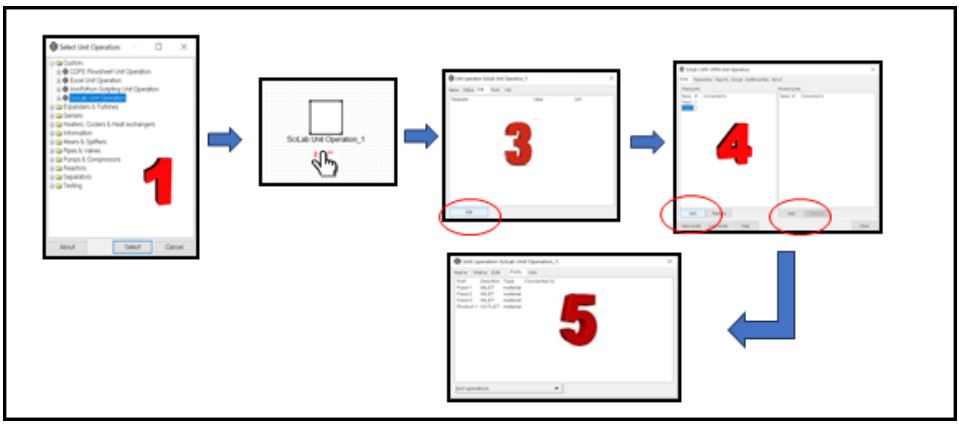
1. COFE untuk flowsheeting engine
2. TEA (Thermodynamics for Engineering Applications) untuk system Thermodinamika
3. COUSCOUS koleksi unit operation (peralatan bidang Teknik Kimia)
4. CORN untuk paket reaksi numerik

Distribusi dari COCO juga selalu menyertakan perangkat lunak ChemSep LITE. Dimana ChemSep merupakan perangkat lunak proses separasi destilasi berdasarkan pada sistem

kesetimbangan. Untuk mendapatkan perangkat lunak ini dapat diunduh pada alamat: http://www.chemsep.com/ atau https://www.cocosimulator.org/ (*CAPE-OPEN*, 2020).

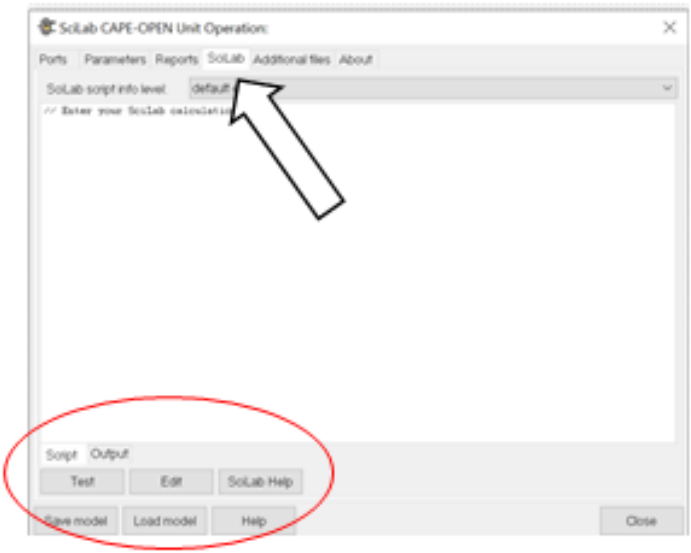
Integrasi dari beberapa program yang tergabung dalam CAPE OPEN ini dengan Scilab, maka perangkat lunak Scilab harus terinstal pada komputer. Program interaksi dikembangkan oleh AMSTERCHEM corporation. Program "Scilab unit operation" yang dapat diunduh pada "https://amsterchem.com/scilabunitop.html" . Untuk pengguna individu dan akademisi dapat menggunakan lisensi non-komersial yang didapat pada situs yang sama. Selain itu untuk mendapatkan properti paket, perlu diinstal pada komputer "Scilab Thermo Import" yang dapat diunduh "https://amsterchem.com/scilabthermo.html".

Interkoneksi antara Scilab Unit Operation dengan stream sebelum (input) dan sesudah perlu diatur terlebih dahulu ***port input*** dan ***output*** dengan tahapan sebagaimana yang dimunculkan pada gambar 3.1. Hal ini disebabkan pada "**Scilab Unit Operation**" umumnya menggunakan port input hanya dua (2) dan port output satu (1).




Gambar 3.1: Tahapan menambah port

Setelah tahapan penyesuaian *port input-output*, kemudian dilakukan penulisan script Scilab pada unit tersebut. Script atau kode Scilab harus dilalukan TEST terlebih dahulu sebelum dijalankan pada "flowsheet". Penulisan program Scilab dengan menekan pada "tab Scilab" sebagaimana ditunjukkan pada gambar 3.2, dimana "script Scilab" ditulis, kemudian untuk memeriksa apakah script tersebut dapat dijalankan dengan benar, maka klik "Test" dan hasilnya akan muncul pada Tab Output.




Gambar 3.2. Script Scilab – Test – Output

## 3. Hasil dan Diskusi

Pada bagian ini akan dibahas mengenai pemodelan matematis yang digunakan untuk menyusun program (*script*) pada **Scilab Unit Operation** (**SUO**).

### 3.1. Mixer

Tujuan mengembangkan model matematis untuk sistem mixer (pencampur) dengan prinsip arus material masuk ke Mixer lebih dari satu menjadi satu arus. Bagian ini merupakan pengembangan dari aplikasi DWSIM sebagaimana dapat dilihat pada alamat berikut:
https://spoken-tutorial.org/watch/DWSIM/Custom+Unit+Operation+using+Scilab/English/

Asumsi: Keadaan *steady state*, pencampuran sempurna (homogen), tidak ada panas yang hilang ke sekitar (adiabatis).
Persamaan (model matematis) yang digunakan:

- Neraca material (mol) total:

$$totF = \sum F \qquad [3.1]$$

- Neraca komponen:

$$F = \sum f_i \qquad [3.2]$$

- Tekanan keluar mixer

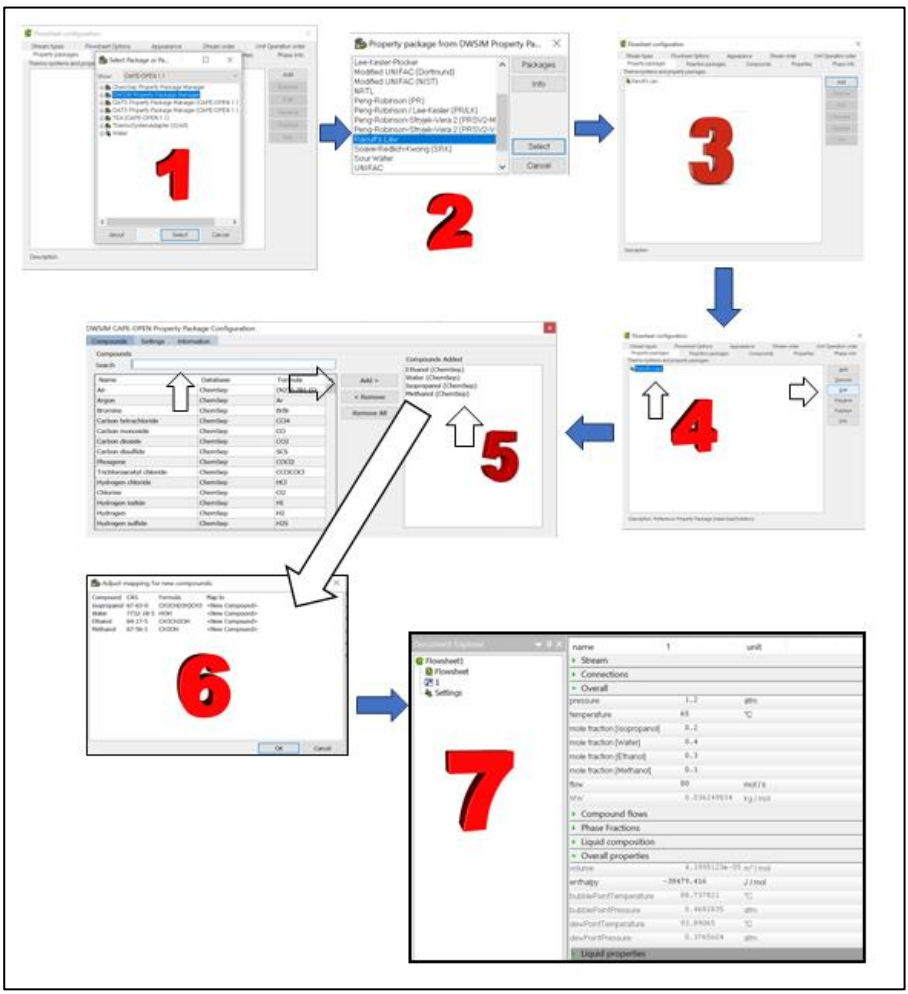
$$P = \left( \frac{\sum p_i}{n} \right) \qquad [3.3]$$

- Neraca energi (entaphi)

$$H = \left( \frac{\sum h_i f_i}{n} \right) \qquad [3.4]$$

Tabel 3.1. Contoh perhitungan Mixer

| Stream / Variabel | Feed 1 | Feed 2 | Feed 3 | Output | Satuan/ unit |
|---|---|---|---|---|---|
| Temperature | 65 | 40 | 30 | ? | $^{o}$C |
| Molar flow | 80 | 190 | 190 | ? | Mol/s |
| Fr. mol Water | 0.4 | 0.2 | 0.1 | ? | - |
| Fr. mol Ethanol | 0.3 | 0.7 | 0.3 | ? | - |
| Fr. mol i-propanol | 0.2 | 0.05 | 0.4 | ? | - |
| Fr. mol methanol | 0.1 | 0.05 | 0.2 | ? | - |
| Tekanan | 1.2 | 1.1 | 1 | ? | Atm |
| Properti paket | Raoult's Law | Raoult's Law | Raoult's Law | | |

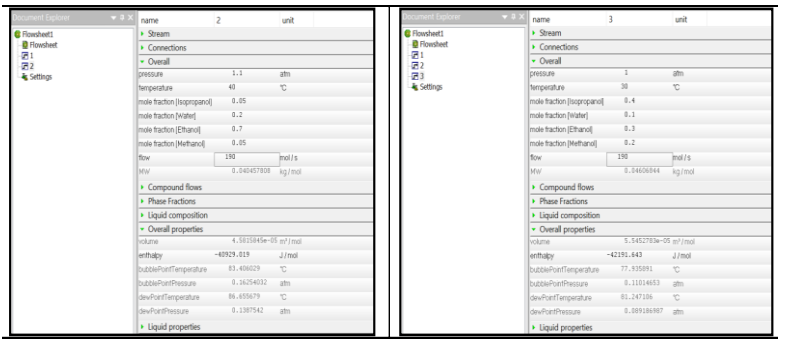



Gambar 3.3. Tahapan pembuatan stream material.

Pada gambar 3.3 merupakan tahapan penyusunan aliran bahan (*material stream*) yang akan masuk pada alat, dimana komposisi, kondisi operasi serta properti paketnya seperti tertera pada tabel 3.1. Dengan menggunakan COCO (CAPE OPEN to CAPE OPEN), maka tahapan mulai dari pemilihan properti paket (tahap 1 sampai dengan 3), dilanjut dengan komponen pada sistem

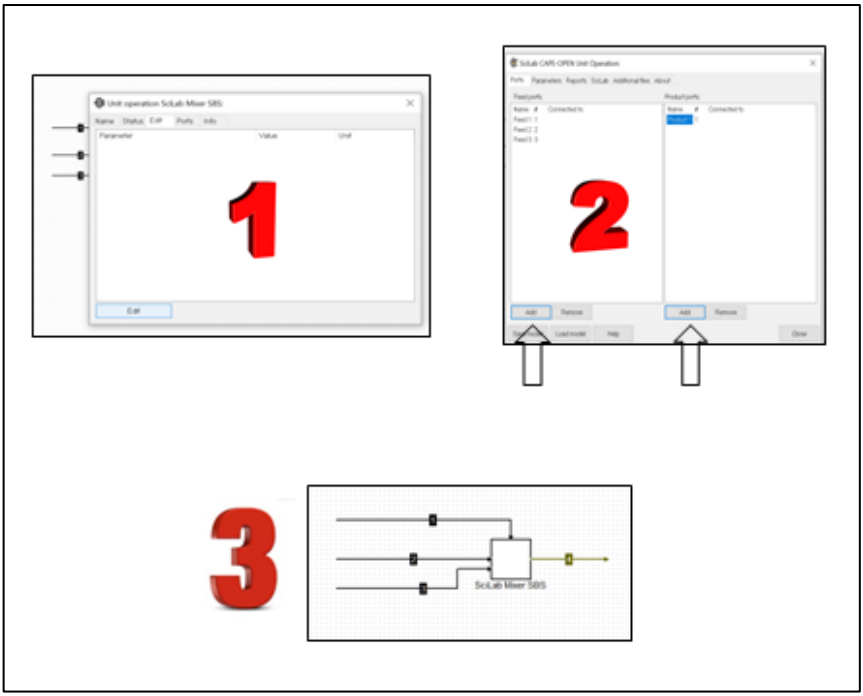
(tahap 4 sampai dengan 6) dan kemudian pengisian kondisi operasi dari aliran (*stream*). Apabila pengisian sudah lengkap, maka perangkat lunak akan menghitung parameter lainnya, seperti enthalpi, dan lainnya sebagaimana ditunjukkan pada tahap 7 gambar 3.3.

Pada bagian ini digunakan properti paketnya DWSIM, meskipun digunakan perangkat lunak COCO. Dengan demikian terlihat ada konektivitas antar perangkat lunak yang terhimpun di COCO. Selanjutnya untuk aliran 2 dan 3 dapat dilihat pada gambar 3.4.



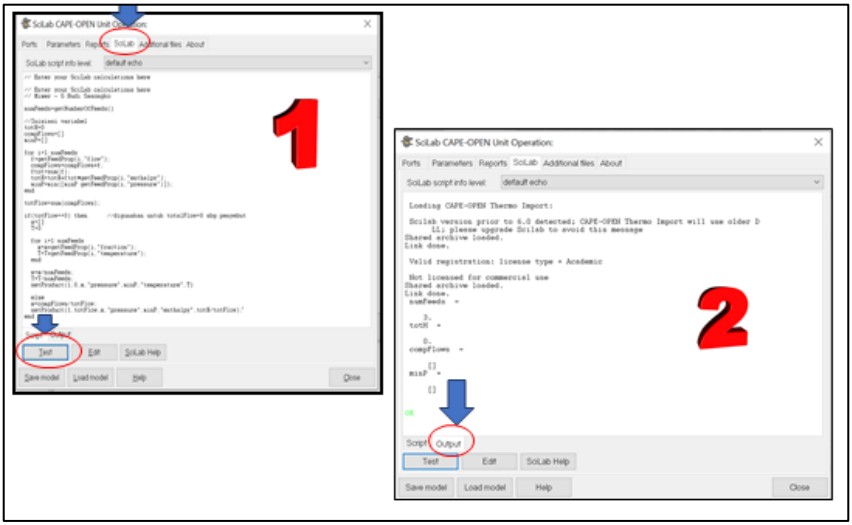

Gambar 3.4. Karakteristik aliran 2 dan 3.

Pada umumnya, Scilab Unit Operation hanya memiliki dua aliran input dan satu aliran output, bahkan aliran input dan output harus di isikan sesuai dengan pengguna. Tanpa pengaturan port Input dan Output, maka aliran material tidak dapat di koneksikan dengan unit. Oleh karenanya perlu dilakukan pengaturan port input dan output. Untuk tahapan pengisian port input dan output dapat dilihat pada gambar 3.5. Pada tahap pertama klik edit pada unit operation, selanjutnya tahap kedua penambah port input dan output sesuai dengan kebutuhan sistem. Tahap ke tiga konenksi antara alat dengan stream.



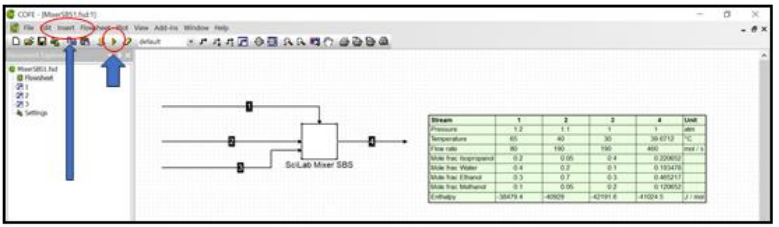

Gambar 3.5. Penambahan aliran Input dan Output pada Scilab Unit Operation

Tahap berikutnya adalah penulisan script Scilab seperti pada gambar 3.6. Setelah penulisan script Scilab selesai, sebelum dieksekusi pada flowsheet harus di TEST dahulu pada Unit, dengan cara klik tombol TEST, hasil eksekusi akan terlihat pada tab "Output". Apabila tidak terjadi kesalahan baik dalam penulisan script maupun logika pemrograman akan didapat keluaran sebagaimana ditunjukkan pada gambar 3.6 tahap 2. Program tersebut dapat disimpan secara parsial dengan menekan tombol "Save model", sehingga unit dapat digunakan kembali untuk aktivitas sistem yang lain dengan menekan tombol "Load model".




Gambar 3.6. Penulisan script Scilab (1), eksekusi – TEST dan tampilan hasil (2).

Apabila pada bagian Unit Operation Scilab sudah tidak ada masalah, kemudian dilanjutkan pada flowsheet yang tergabungan dengan unit lainnya atau sudah dilengkapi dengan stream input maupun output sebagaimana ditunjukkan pada gambar 3.7. Untuk mengetahui apakah sudah tidak ada masalah dengan koneksi, paramter unit maka klik tombol “Validate (F8)”, apabila sudah tidak ada masalah dilanjutkan dengan klik tombol “Solve (F5)”. Hasil dapat dimunculkan dengan cara klik “Insert” → “Unit report …”, maka hasil eksekusi akan tampak sebagaimana pada gambar 3.7.



Gambar 3.7. Eksekusi hasil pada Flowsheet dengan menekan tombol “Validate”, dan “Solve”.

Tabel 3.2. Script program Mixer - Scilab

| No. | Script program - Scilab | Keterangan |
|---|---|---|
| 1 | numFeeds=**getNumberOfFeeds**() | Untuk mendapatkan jumlah feed yang masuk (port) |
| 2 | totH=0<br>compFlows=[]<br>minP=[] | Inisiasi variabel |
| 3 | for i=1:numFeeds<br>f=**getFeedProp**(i,"flow")<br>compFlows=compFlows+f<br>ftot=sum(f)<br>totH=totH+ftot*getFeedProp(i,"enthalpy")<br>minP=min([minP **getFeedProp**(i,"pressure")])<br>end | Looping penjumlahan variable:<br>$F = \sum f_i$<br>$H = \left( \frac{\sum h_i f_i}{n} \right)$ |
| 4 | totFlow=**sum**(compFlows) | $totF = \sum F$ |
| 5 | if(totFlow==0) then<br>x=[]<br>T=0 | Digunakan untuk totalFlow=0 sbg |
| 6 | for i=1:numFeeds | |

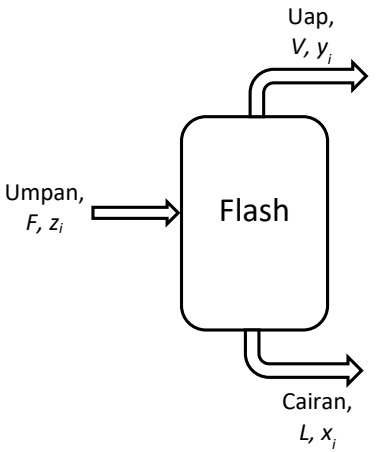
| | | |
|---|---|---|
| | x=x+getFeedProp(i,"fraction")<br>T=T+getFeedProp(i,"temperature")<br>end | penyebut (dihindari) |
| 7 | x=x/numFeeds<br>T=T/numFeeds<br><br>setProduct(1,0,x,"pressure",minP,"temperature",T) | |
| 8 | else<br>x=compFlows/totFlow<br><br>setProduct(1,totFlow,x,"pressure",minP,"enthalpy",totH/totFlow)<br>end | |

Pada tabel 3.1 merupakan script untuk mixer yang dapat digunakan secara umum. Pada tahap satu (1) adalah untuk mengetahui berapa banyak input pada unit. Tahap kedua merupakan inisiasi dari varibel. Mixer merupakan penjumlahan dari semua variable yang masuk, maka tahap ketiga dan empat merupakan inti dari perhitungan mixer. Pada tahap kelima dan selanjutnya digunakan apabila jumlah aliran (total flow) sama dengan nol, hal ini digunakan untuk menghindari pembagian dengan bilangan nol.

### 3.2. Flash Adiabatis

Unit Operasi dari Flash merupakan operasi satu tahapan dimana umpan (multi fase) di flash pada suhu dan/ atau tekanan tertentu dan menghasilkan fase separasi. Pada bagian ini, hanya akan dibahas keluaran dari flash terdiri dari dua fase, yaitu uap dan cairan. Untuk keluaran cairan-cairan atau keluaran multifase

Uap-Cairan-Cairan tidak dibahas. Gambar 3.8 merupakan diagram kesetimbangan flash.




Gambar 3.8. Diagram kestimbangan flash

Pada gambar 3.8 diasumsikan keluaran dari flash dalam keadaan keseimbangan uap-cairan. Berikut persamaan kesetimbangan untuk unit flash.

Persamaan (model matematis) yang digunakan:

- Neraca material total:

$$V + L - F = 0 \qquad [3.5]$$

- Neraca material komponen:

$$V\ y_i + L\ x_i - F\ z_i = 0 \qquad [3.6]$$

- Kesetimbangan uap-cairan:

$$K_i\ x_i - y_i = 0 \qquad [3.7]$$

- Penjumlahan fraksi komponen:

$$\sum_{i=1}^{c}(y_i - x_i) = 0 \qquad [3.8]$$

- Neraca panas (enthalpy):

$$V\,H^V + L\,H^L - F\,H^F + Q = 0 \qquad [3.9]$$

Pada persamaan diatas *F* adalah kecepatan aliran molar umpan untuk fraksi mol komponen $z_i$. Sedangkan *V* dan *L* merupakan kecepatan aliran Uap dan Cairan yang keluar dari *Flash* dengan fraksi mol komponen $y_i$ dan $x_i$.

Rasio kesetimbangan $K_i$ dan enthalpy *H* dihitung dari properti model. Sedangkan *Q* merupakan panas yang ditambahkan pada umpan sebelum flash.

Sebagai bagian validasi dari program maka digunakan contoh yang diambil dari buku Seader halaman 47 (Seader and Henley, 2006), bagian ini merupakan modifikasi dari Arvind Prasad.

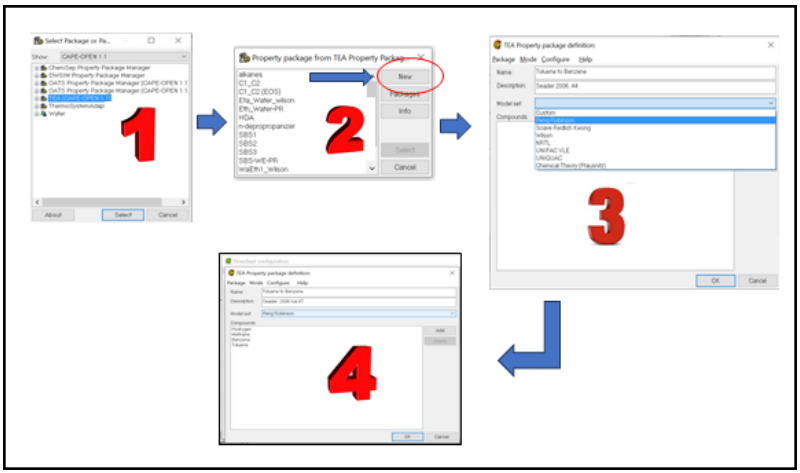
Reaksi Hydrodealkylation termal dari Toluene menjadi Benzene ($C_7H_8 + H_2 \rightarrow C_6H_6 + CH_4$) dilakukan pada tekanan dan suhu tinggi. Pada reaksi ini digunakan Hydrogen berlebihan untuk meminimasi cracking senyawa aromatis menjadi gas ringan, maka produk dari reaktor di separasi dan Hydrogen di recycle kembali. Efluen uap reaktor dengan kecepatan aliran 5.597 kmol/h pada tekanan 500 psia (3.448 kPa) dan suhu 275$^o$F (408,2 K), kemudian didinginkan pada suhu 120$^o$F (322 K) dan dimasukan ke flash drum untuk dipisahkan berdasarkan kesetimbangan Uap-Cair. Tabel 3.2 merupakan kondisi keluar reaktor.

Tabel 3.3. Kondisi keluar reaktor Hydrodealkylasi Toluene menjadi Benzene (Seader, 2006, 44)

| Variabel | Nilai | Satuan |
|---|---|---|
| Tekanan | 3448 | kPa |
| Suhu | 408.2 | K |
| Kecepatan alir | 5.597 | Kmol/h |
| Hydrogen | 0.3177 | Fraksi mol |
| Methane | 0.5894 | Fraksi mol |
| Benzene | 0.0715 | Fraksi mol |
| Toluene | 0.0214 | Fraksi mol |

Hitung komposisi kesetimbangan efluen dari reaktor, kecepatan aliran uap dan cairan serta panas yang harus ditranfer pada sistem. Sebagai properti paket akan digunakan **Peng-Robinson**.

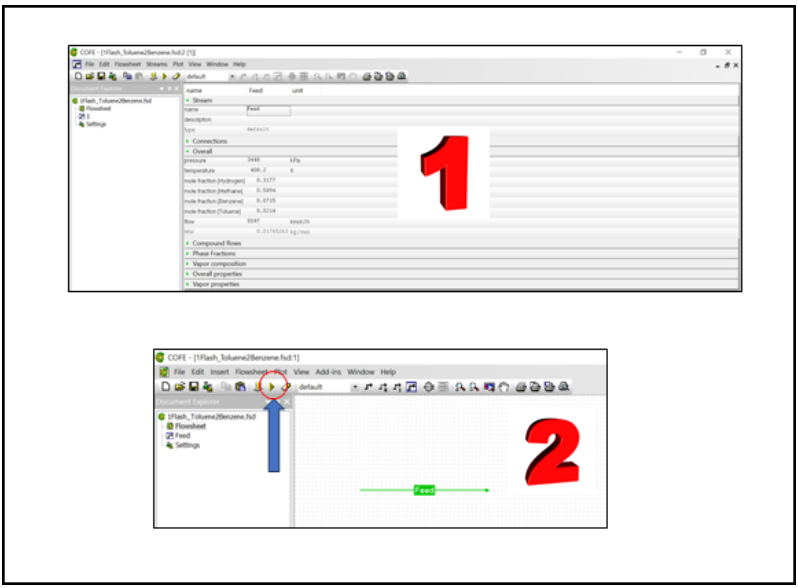
Untuk membuat menyelesaikan dari permasalahan diatas, akan disimulasikan terlebih dahulu akan digunakan COCO simulator.




Gambar 3.9. Tahapan pemilihan **properti paket** dan **komponen**

Untuk menyelesaikan menggunaka COCO Simulator, tahapan awal adalah menentukan properti paket dengan cara klik **setting** yang terletak disebelah kiri. Pada bagian ini akan digunakan paket manager dari **TEA (CAPE-OPEN 1.1)**. Tahapan kedua (**2**) dari Gambar 3.9 membuat properti pkaet dan komponen baru dengan klik **New**, selanjutnya tahap ketiga (**3**) adalah membuat nama yang akan disimpan, mendeskripsikan, memilih **properti paket** yang sesuai, setelah pengisian lengkap (semua komponen yang akan digunakan sudah dipilih), maka tampilan 4 akan muncul dan klik **OK**.

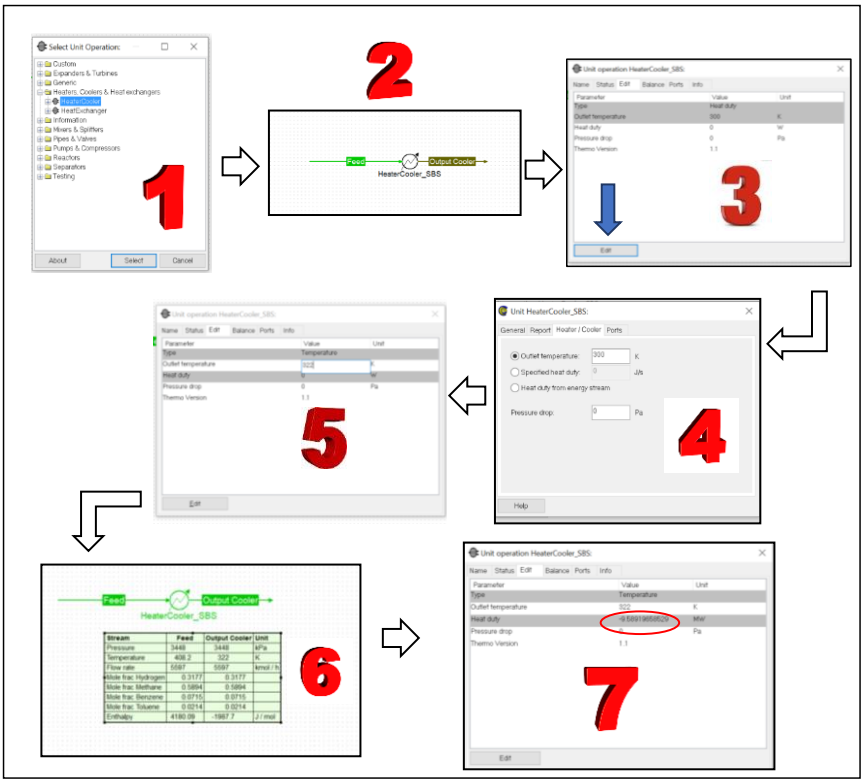
Selanjutnya dibuat material stream pada flowsheet, klik dua kali pada stream, kemudian diisikan sesuai dengan kondisi sebagaimana pada tabel 3.2, sebagaimana ditunjukkan pada gambar 3.10. Kemudian klik **Solve (F5)**, untuk menyelesaikan pada stream material tersebut, sampai warna pada garis berubah menjadi hijau.




Gambar 3.10. Tahapan penyusunan **Material Stream**.

Pada simulasi ini akan dilakukan penurunan suhu terlebih dahulu sebelum masuk pada unit flash. Tujuan dengan penambahan unit ini adalah untuk mengetahui berapa banyak energi yang diperlukan untuk menurunkan suhu dari 408.2 K menjadi 322 K.

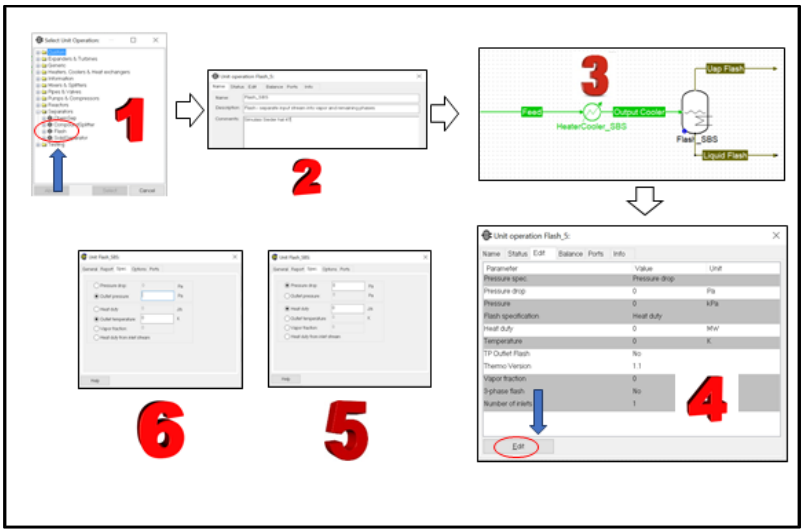
Oleh karenanya ditambah unit "**HeaterCooler**", sebagaimana ditunjukkan pada gambar 3.11. Kemudian pada tahap kedua (2) ditambah stream keluar dari unit "HeaterCooler". Tahap selanjutnya (ketiga) adalah mengatur kondisi dari unit "HeaterCooler". Pada gambar 3.11 terlihat pada unit tersebut sebagai *default* yang dapat diatur adalah "*Heat duty*" artinya jumlah panas yang dimasukkan, sedangkan yang akan diatur adalah suhu keluar dari unit, oleh karenanya perlu diubah dengan cara klik tombol "**Edit**" pada bagian bawah, kemudian klik pada *radio button* "**Outlet Temperature**" (tahap 4), tutup jendela dan isikan suhu keluarkan yang akan disimulasikan yaitu 322 K. Kemudian **Solve (F5)** pada sistem sampai semua berwarna hijau. Hasil simulasi dari unit ini adalah energi yang diperlukan untuk menurunkan suhu dari 408.2 K menjadi 322 K adalah sebesar – 9.589 MW (tanda minus energi yang dikeluarkan atau pendinginan).




Gambar 3.11. Penambahan dan simulasi unit "**HeaterCooler**"

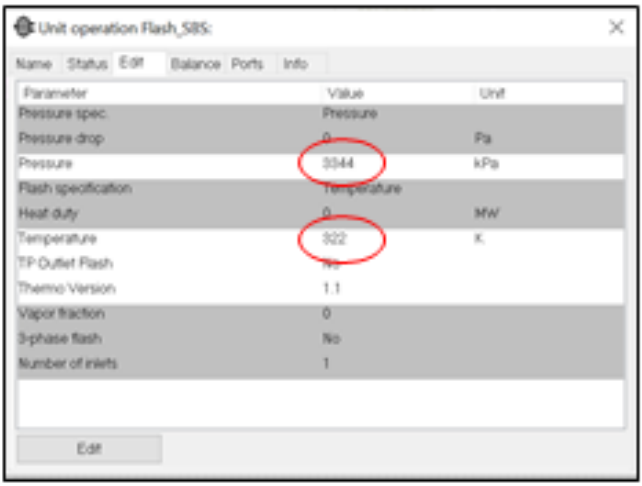
Tahap selanjutnya setelah simulasi dengan "HeaterCooler" berhasil dilanjutkan dengan menambahkan unit **flash** sebagaimana ditunjukkan pada gambar 3.12. Tahap satu (1) dan dua (2) adalah menambahkan unit dan memberi nama. Tahap ketiga (3) menggabungkan stream material keluar dari unit "HeaterCooler" pada unit "Flash" serta menambah stream output berupa stream "Uap Flash" dan "Liquid Flash". Selanjutnya mengatur unit "Flash" dengan cara double klik pada unit. Pada gambar 3.12 tahap keempat (4) terlihat parameter yang dapat diatur pada unit (yang bukan abu-abu). Sedangkan yang akan disimulasikan adalah tekanan flah drum sebesar 3344 kPa dan

suhu flash sebesar 322 K. Oleh karenanya pada unit harus diatur dengan klik **Edit**, maka akan muncul jendela sebagaimana pada gambar (tahap) 5. Untuk merubah klik pada Radio Button "Output Pressure" dan "Output Temperature" sebagaimana pada tahap 6.



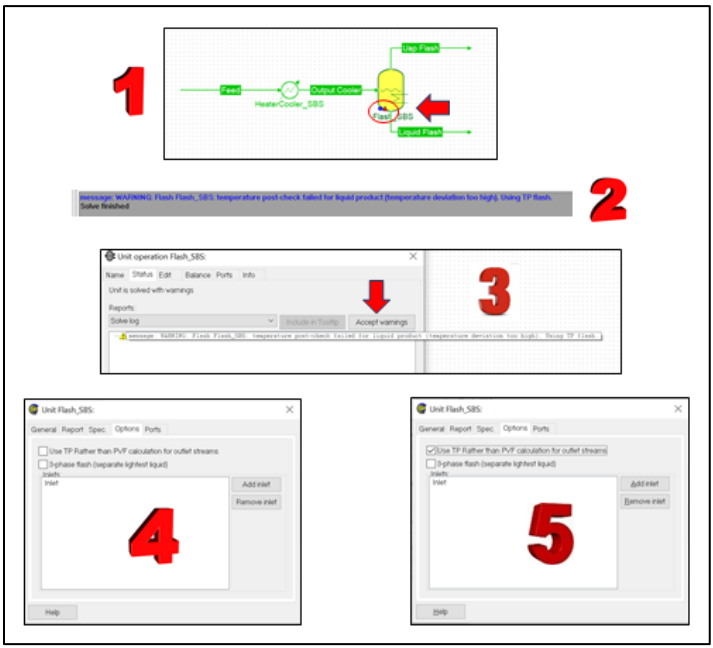

Gambar 3.12. Penambahan dan simulasi unit "**Flash**"

Setelah penambahan unit flash sebagaimana tahapan yang harus dieksekusi seperti pada gambar 3.12, selanjutnya simulasi dengan memasukkan parameter yang dieksekusi yaitu tekanan flah drum sebesar 3344 kPa dan suhu flash sebesar 322 K. Gambar 3.13 merupakan yang diinputkan sebagai parameter yang diinginkan. Selanjutnya "solve (F5)" sistem, hasil dari simulasi sebagaimana terlihat pada gambar 3.14.

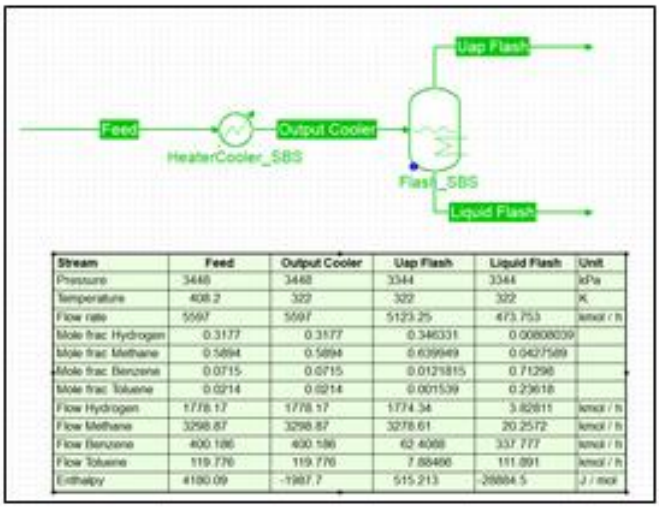



Gambar 3.13. Input parameter simulasi “Flash”

Pada gambar 3.14 terlihat pada unit “Flash” simulasi tidak bisa diselesaikan dan ada *message* (poin 2) yang muncul yaitu kesalahan *temperature post-check* pada produk cairan disarankan menggunakan “TP Flash”. Saran diterima dengan klik “Accept warning” (poin 3). Kemudian klik pada Edit dan beri centang pada “*Use TP rather than PVF calculation for Outlet Stream*” (poin 5). Kemudian klik “*Solve F5*” kembali dan hasil simulasi berhasil (semua hijau) sebagaimana gambar 3.15.



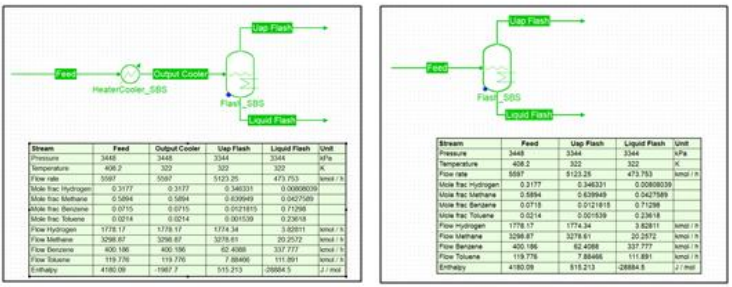

Gambar 3.14. Proses mengatasi masalah pada simulasi.




| Stream | Feed | Output Cooler | Uap Flash | Liquid Flash | Unit |
|---|---|---|---|---|---|
| Pressure | 3448 | 3448 | 3344 | 3344 | kPa |
| Temperature | 408.2 | 322 | 322 | 322 | K |
| Flow rate | 5597 | 5597 | 5123.25 | 473.753 | kmol / h |
| Mole frac Hydrogen | 0.3177 | 0.3177 | 0.346331 | 0.00808039 | |
| Mole frac Methane | 0.5894 | 0.5894 | 0.639849 | 0.0427589 | |
| Mole frac Benzene | 0.0715 | 0.0715 | 0.0121815 | 0.71298 | |
| Mole frac Toluene | 0.0214 | 0.0214 | 0.001539 | 0.23618 | |
| Flow Hydrogen | 1778.17 | 1778.17 | 1774.34 | 3.82811 | kmol / h |
| Flow Methane | 3298.87 | 3298.87 | 3278.61 | 20.2572 | kmol / h |
| Flow Benzene | 400.186 | 400.186 | 62.4088 | 337.777 | kmol / h |
| Flow Toluene | 119.776 | 119.776 | 7.88486 | 111.891 | kmol / h |
| Enthalpy | -180.09 | -198.7 | 515.213 | -26884.5 | J / mol |

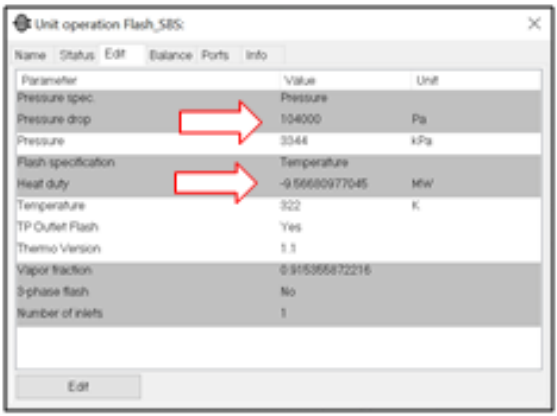
Gambar 3.15. Hasil simulasi Flash

Simulasi tahap selanjutnya adalah unit HeaterCooler dihilangkan, yang selanjutnya unit tersebut akan masuk pada unit Flash. Sebelum diubah, maka sebaiknya "*safe as*" dan disimpan dengan nama yang lain, misalkan dengan angka. Kemudian unit HeaterCooler dihilangkan dan material stream Feed masuk ke unit Flash, run atau "solve F5" simulasi. Hasil keduanya diperbandingkan sebagaimana ditunjukkan pada gambar 3.16.



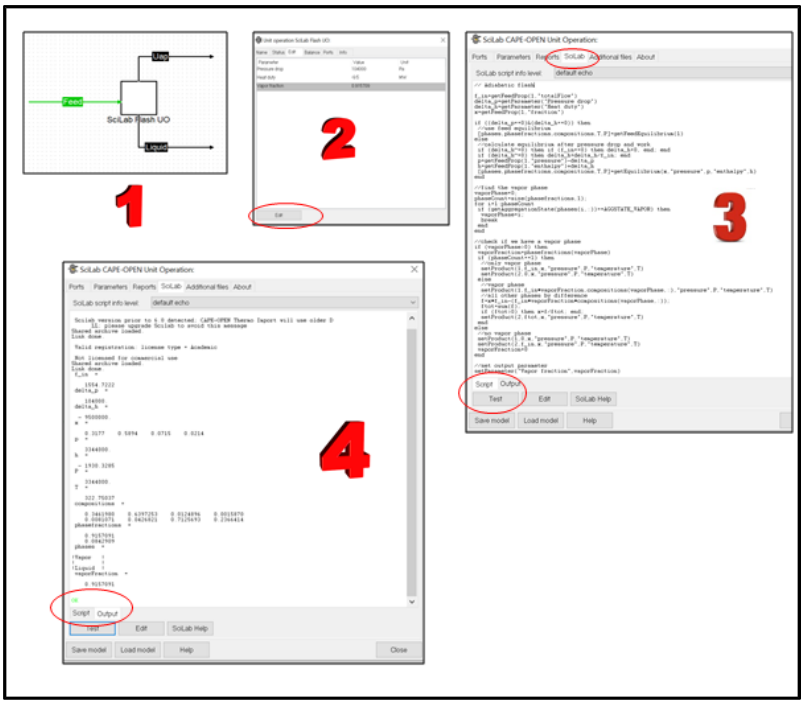

Gambar 3.16. Perbandingan Unit flash dengan dan tanpa unit Cooler.

Pada gambar 3.17 terlihat hasil simulasi tanpa cooler, dengan menetapkan tekanan 3344 kPa dan suhu flash 322 K, maka pressure drop sebesar 104000 Pa dan Heat duty sebesar – 9.567 MW.



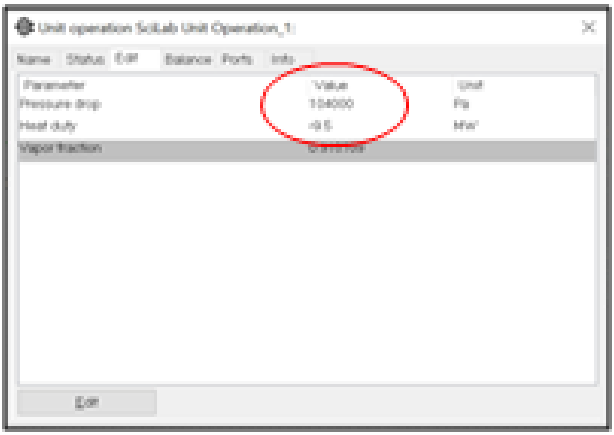

Gambar 3.17. Hasil simulasi tanpa Cooler, (Presure drop dan Heat duty)

Simulasi **ketiga** adalah mengganti flash dengan Scilab CAPE-OPEN Unit Operation. Pada bagian ini, Unit Operation di *create* dengan menggunakan Scilab dan prosedur penyusunan secara garis besar seperti pada sub-bab 3.1. Mula-mula membuat *stream material 1* (masuk), seperti pada simulasi kesatu dan simulasi kedua. Kemudian solve-F5, selanjutnya tambahkan "Scilab CAPE-OPEN Unit Operation", atur Port masuk dan Port keluar. Dilanjutkan dengan mengkoneksikan stream masuk dan dua stream keluar berupa Uap (2) dan Liquid (3). Kemudian double klik pada "Scilab flash UO", dilanjutkan dengan klik Edit pada bagian bawah. Selanjutnya Script Scilab ditulis, kemudian klik Test, apabila tidak ada kesalahan maka pada Output akan berakhir dengan OK. Tahapan penulisan Scilab Falsh UO dapat dilihat pada gambar 3.18.

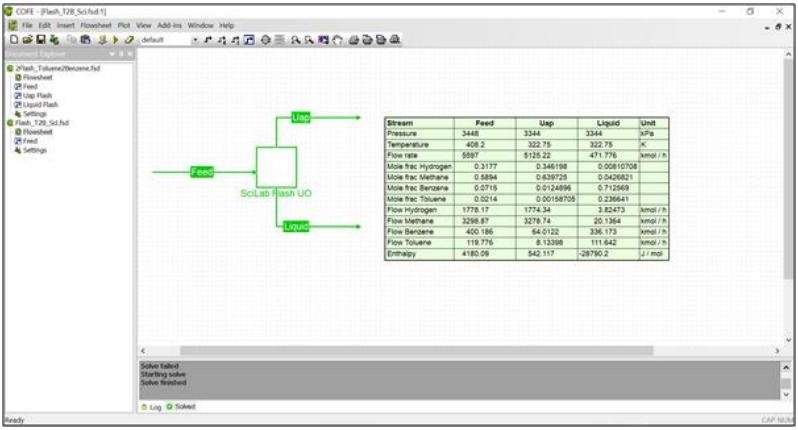



Gambar 3.18. Tahapan penulisan script pada Scilab-flash Unit Operation.

Setelah proses penulisan script Scilab dan Test dari Script sudah dilakukan dan berhasil, tahap selanjutnya adalah Simulasi. Pada tahap ini diperlukan Parameter “Pressure drop” dan “Heat duty”. Nilai ini didapat dari simulasi sebelum dimana nilainya dapat dilihat pada gambar 3.17. Nilai tersebut dimasukkan sebagaimana ditunjukkan pada gambar 3.19. Sedangkan hasil simulasi dapat dilihat pada gambar 3.20.



Gambar 3.19. Nilai parameter untuk Flash Scilab.




Gambar 3.20. Hasil simulasi Scilab Flash Unit Operation

Tabel 3.4. Perbandingan simulasi dari Sieder, Coco simulator dan Scilab Flash UO.

| Variabel | Sieder, 2006 | Coco simulator | Scilab Unit Operation |
|---|---|---|---|
| Aliran Uap, kmol/h | | | |
| Hydrogen | 1777.8 | 1774.34 | 1774.34 |
| Methane | 3278.5 | 3278.61 | 3278.74 |
| Benzene | 61.9 | 62.4088 | 64.0112 |
| Toluene | 7.4 | 7.88466 | 8.13398 |
| Total | 5122.7 | 5153.25 | 5125.22 |
| Aliran Cairan, kmol/h | | | |
| Hydrogen | 3.3 | 3.8281 | 3.8247 |
| Methane | 20.4 | 20.2572 | 20.1364 |
| Benzene | 338.2 | 337.777 | 336.173 |
| Toluene | 112.4 | 111.891 | 111.642 |
| Total | 474.3 | 473.753 | 471.776 |

Pada tabel 3.3 merupakan perbandingan antara simulasi yang dilakukan oleh Sieder, 2006 dengan Coco simulator dan Scilab Unit Operation. Pada tebel terlihat bahwa nilai hasil simulasi tidak ada yang eksak sama. Hal ini disebabkan pada perhitungan secara numeris dari ketiga sistem tersebut tidak sama persis. Hal ini dapat dilihat dari Kolom ke tiga dan kolom ke empat, meskipun dikerjakan dengan perangkat yang sama tetapi hasil perhitungan agak berbeda.

Script dari perhitungan Scilab Flash dapat dilihat pada tabel 3.4. Pada script terlihat bahwa Scilab UO sudah dilengkapi dengan sub rutin untuk perhitungan kesetimbangan yang dapat digunakan pada perangkat lunak berbasis CAPE-OPEN.

Tabel 3.5. Script program Flash - Scilab

| No. | Script program - Scilab |
|---|---|
| 1 | f_in=getFeedProp(1,"totalFlow")<br>delta_p=getParameter("Pressure drop")<br>delta_h=getParameter("Heat duty")<br>x=getFeedProp(1,"fraction") |
| 2 | if ((delta_p==0)&(delta_h==0)) then<br>//use feed equilibrium<br>[phases,phasefractions,compositions,T,P]=getFeedEquilibrium(1)<br>else<br>//calculate equilibrium after pressure drop and work<br>if (delta_h~=0) then if (f_in==0) then delta_h=0, end; end<br>if (delta_h~=0) then delta_h=delta_h/f_in; end<br>p=getFeedProp(1,"pressure")-delta_p<br>h=getFeedProp(1,"enthalpy")+delta_h<br>[phases,phasefractions,compositions,T,P]=getEquilibrium(x,"pressure",p,"enthalpy",h)<br>end |
| 3 | //find the vapor phase<br>vaporPhase=0;<br>phaseCount=size(phasefractions,1);<br>for i=1:phaseCount<br>if (getAggregationState(phases(i,:))==AGGSTATE_VAPOR) then<br>vaporPhase=i;<br>break<br>end<br>end |
| 4 | //check if a vapor phase<br>if (vaporPhase>0) then<br>vaporFraction=phasefractions(vaporPhase) |

```
 if (phaseCount==1) then
  //only vapor phase
  setProduct(1,f_in,x,"pressure",P,"temperature",T)
  setProduct(2,0,x,"pressure",P,"temperature",T)
 else
  //vapor phase
setProduct(1,f_in*vaporFraction,compositions(vaporPhase,
:),"pressure",P,"temperature",T)
  //all other phases by difference
  f=x*f_in-
(f_in*vaporFraction*compositions(vaporPhase,:));
  ftot=sum(f);
  if (ftot>0) then x=f/ftot; end;
  setProduct(2,ftot,x,"pressure",P,"temperature",T)
 end
else
 //no vapor phase
 setProduct(1,0,x,"pressure",P,"temperature",T)
 setProduct(2,f_in,x,"pressure",P,"temperature",T)
 vaporFraction=0
end
```

```
5  //set output parameter
   setParameter("Vapor fraction",vaporFraction)
```

Pada baris pertama merupakan perintah untuk mendapatkan properti pada Feed dari unit dalam hal ini *Flash*, seperti Aliran total, komposisi, panas, maupun *pressure drop*. Pada baris kedua untuk menghitung kesetimbangan pada Feed. Baris tiga untuk menghitung fase uap, sedangkan baris ke empat menghitung kesetimbangan pada unit flash dan hasilnya dimunculkan pada baris kelima.

## 4. Kesimpulan

Penyusunan script Scilab pada unit COCO Simulator dapat berfungsi dengan baik. Penerapan aplikasi ini dapat digunakan untuk pengembangan perangkat lunak pada unit operation yang bersifat spesifik sehingga karakteristik *blackbox* pada sistem simulator dapat diubah menjadi sistem *whitebox* dengan model ini.

## 5. Referensi


Baten, Van, J., Taylor, R. and Kooijman, H. (2010) 'Using Chemsep, COCO and modeling tools for versatility in custom process modeling', *AIChE annual meeting*. Available at: http://www.cocosimulator.org/downloads/SaltlakeCityAIChE2010ExtendedAbstract_Chemsep_COCO.pdf.

*CAPE-OPEN* (2020). Available at: http://www.cocosimulator.org/.

Seader, D. and Henley, E. J. (2006) *SEPARATION PROCESS PRINCIPLES*. John Wiley & Sons, Ltd.

# BAB IV

# Mengenal Simulator CAPE-OPEN-COCO

Setia Budi Sasongko (sbudisas@live.undip.ac.id)

**Abstraksi:**

Pada bagian ini dibahas mengenai penggunaan perangkat lunak COCO secara detail, artinya dengan mengikuti tahapan-tahapan yang ada diharapkan dapat pembaca dapat menggunakan perangkat lunak simulator COCO dengan baik dan benar.

Dua bagian besar sebagai ilustrasi yang dikembangkan pada bab ini, pertama penggunaan COCO tanpa reaksi kimia dan yang kedua dengan adanya reaksi kimia.

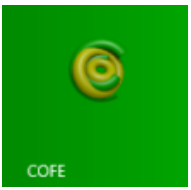
Perangkat lunak ini dapat diunduh secara gratis pada alamat: https://www.cocosimulator.org/index.html.

## 1. Pendahuluan

**CAPE-OPEN** merupakan kepanjangan dari Computer-Aided Process Engineering Open Simulation Environment merupakan proyek kolaborasi antara grup operating companies, technology vendors dan grup akademik. Pada bagian ini akan digunakan **COCO** atau **CAPE-OPEN to CAPE-OPEN** merupakan piranti lunak simulasi bebas bayar alias gratis yang mempunyai compatibilitas cukup tinggi terhadap CAPE-OPEN yang lain. COCO terdiri dari empat bagian yaitu:

- **COFE**: CAPE-OPEN untuk Flow-sheeting
- **TEA**: Termodinamika untuk aplikasi keteknikan

- **COUSCOUS**: CAPE-OPEN untuk Unit-Operation (sederhana)
- **CORN**: CAPE-OPEN untuk reaksi numerik.

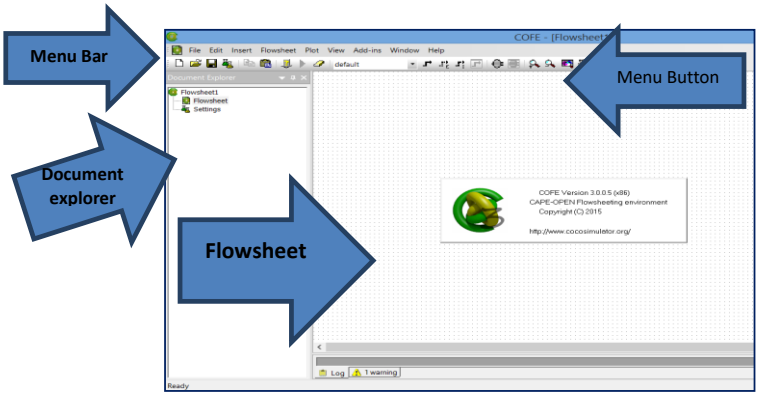



Secara garis besar dalam menyusun simulasi COCO adalah:

1. Menentukan Property Pack (misalkan TEA)
2. Menentukan “Thermodynamic model set”
3. Menyusun komponen yang akan digunakan dalam simulasi
4. Menentukan Unit Operation
5. Menentukan kondisi operasi
6. Simulasi
7. Tampilkan hasil

Pada bagian lampiran ini, akan dibahas secara detail dengan menggunakan gambar setiap tahapan, dengan harapan dapat diikuti dengan baik.

Tampilan awal dari COFE,sebagaimana ditunjukkan pada gambar 4.1, dimana pada bagian atas terlihat menu bar yang berisi menu-menu perintah dalam dalam bentuk tulisan, pada baris selanjutnya adalah menu button, merupakan menu perintah dalam bentuk symbol gambar, sebelah kiri adalah *document explorer*, dan sebelah kanannya tempat *flowsheet*.



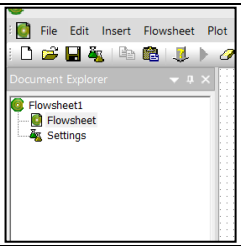
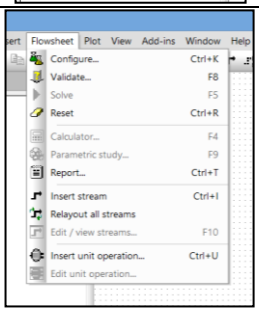
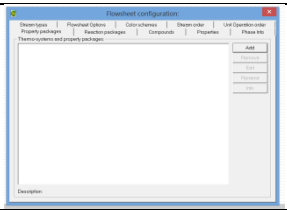

Gambar 4.1. Tampilan awal COFE

## 2. ILUSTRASI FLASH DRUM – Ethanol - Air

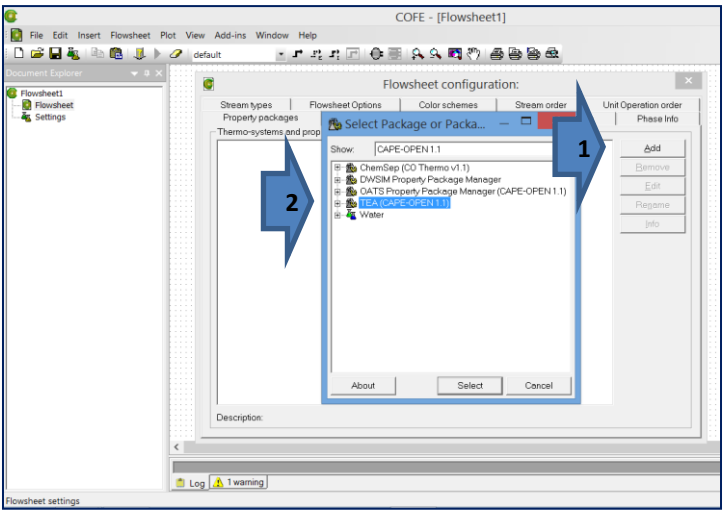
Untuk mempermudah dalam pemahaman dan dapat langsung dipraktekan, maka perlu ilustrasi.

Campuran ethanol dan air ($H_2O$) dengan menggunakan model thermodinamika "Peng-Robinson" dimasukan pada alat separator "Flash" pada kondisi operasi tekanan 1 bar, suhu 93.5$^{o}$C dengan komposisi 139 mol/det air ($H_2O$) dan 26 mol/det ethanol.

**2.1. Menentukan property pack dapat dilakukan dengan 3 cara (pilih salah satu)**

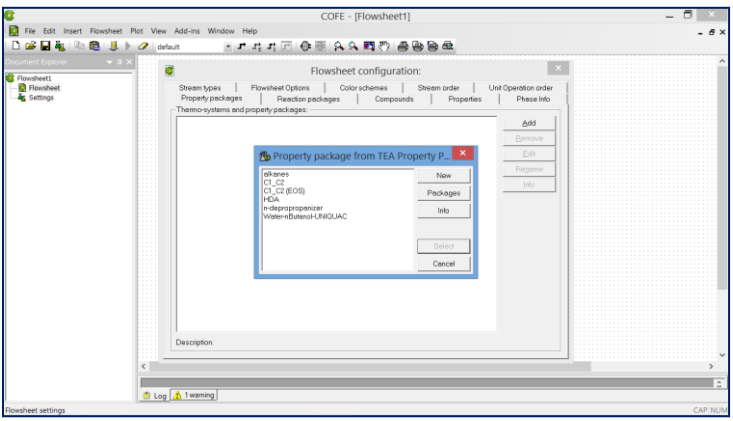
| No. | Perintah | Tampilan |
|---|---|---|
| a. | Klik pada “Document Explorer” flowsheet1 → Settings |  |
| b. | Pada tools bar: klik “Flowsheet” → Configure |  |
| No. | Perintah | Tampilan |
| c. | Cntrl + K |  |

- Klik **Add:** (misal) pilih TEA (CAPE-OPEN 1.1)




Gambar 4.2. Pemilihan Property Package

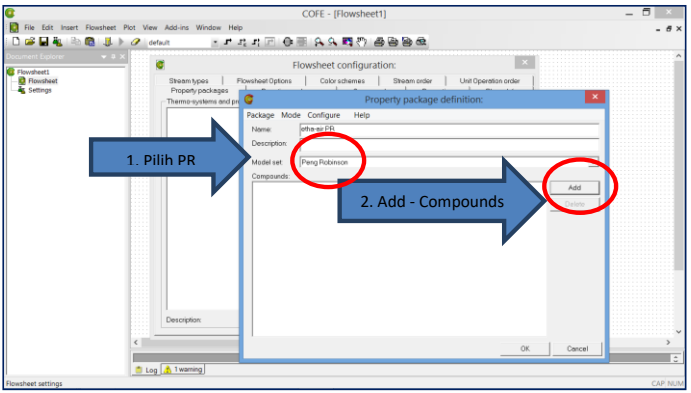
- Karena property package yang ada tidak sesuai dengan yang akan digunakan, maka Klik **New** - beri nama dan deskripsi model




Gambar 4.3. Pembuatan PP baru

## 2.2. Menentukan “Thermodynamic model set”

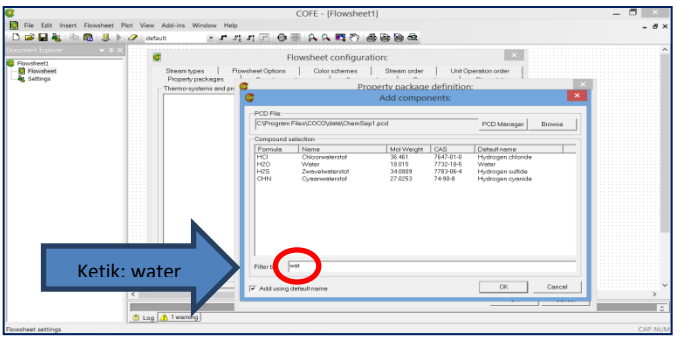
- Model set: - pilih **Peng-Robinson**




Gambar 4.4. Mendefinisikan Property Package

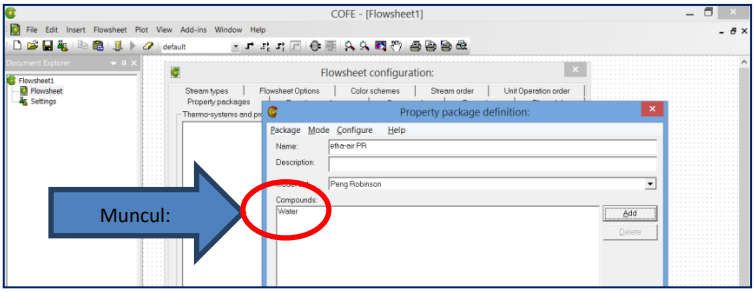
## 2.3. Menyusun komponen yang akan digunakan dalam simulasi

- Pada isian **Filter by** – ketik water – sorot **H2O** atau **Water** – double klik atau klik tombol OK.




Gambar 4.5. Pemilihan Komponen

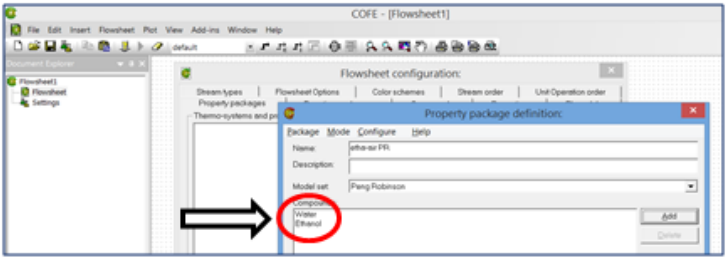
Selanjutnya pada bagian “*Compounds*” akan muncul komponen “*water*”.




Gambar 4.6. Senyawa (Water) berhasil dipilih.

- Pengerjakan yang sama untuk komponen: **Ethanol**.

Dengan hasil akhir pada komponen: **Air dan Ethanol**.



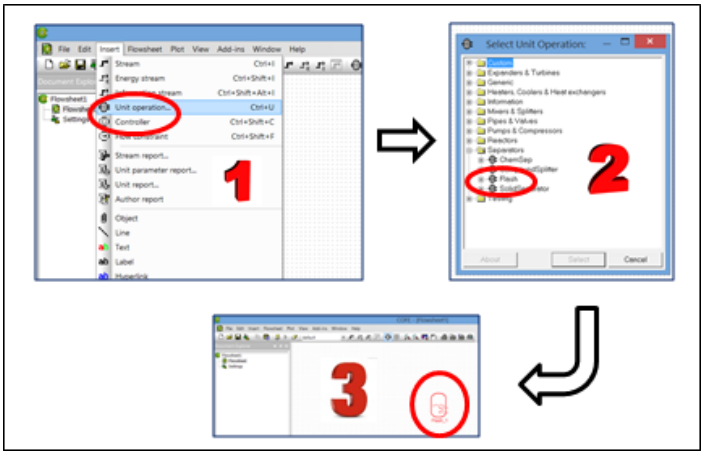

Gambar 4.7. Komponen Water-Ethanol yang dipilih pada sistem

## 2.4. Menambahkan Unit Operation

Selanjutnya akan ditambahkan Unit Operation pada flowsheet.

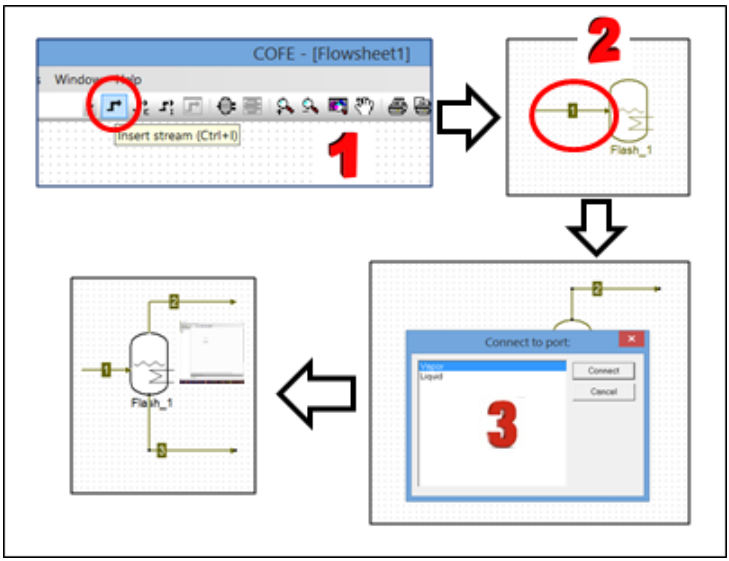
- Klik Flowsheet
- Menu Insert → pilih Unit Operation
- Pada Jendela: Unit Operation → Separators → Flash → klik Tombol **Select**
- Maka akan mucul Simbol Flash → letakan di area flowshet
- Jika akan mengganti nama, Klik kanan → pilih **rename**

Tahapan dari penambahan alat Unit Operation ini dapat dilihat pada gambar 4.8.




Gambar 4.8. Penambahan alat – *Unit Operation*.

- Menambahkan **stream** (aliran) dari Unit Operation dengan meng-klik **menu button**: tombol **Insert Stream** (sebagaimana di gambar) **atau** melalui **menu bar**: Insert → pilih Stream.
- Selanjutnya cursor mouse akan berubah menjadi tanda plus (+)
- Klik pada flowsheet pada bagian kiri dari flash drum kemudian koneksikan dengan membawa mouse pada bagian samping kiri dari flash drum klik mouse kanan, maka akan muncul nomor aliran sesuai dengan urutannya. Apabila akan mengganti dapat dilaksanakan dengan klik mouse kanan.




Gambar 4.9. Penambahan *stream* (Aliran) pada Unit Operation.

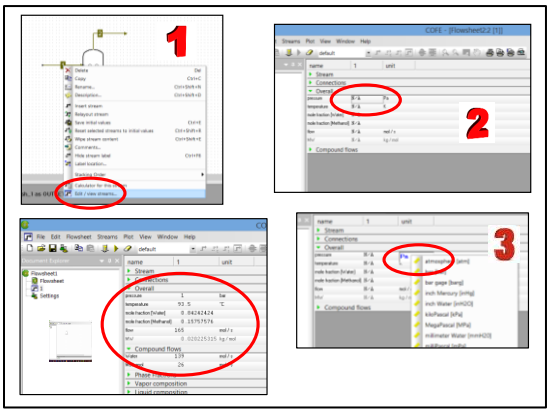
- Klik kembali untuk menambahkan aliran keluar, klik pada bagian atas dari flash drum kemudian mouse ditarik keatas dan ke kanan, klik pada bagian akhir dari aliran.
- Perangkat akan menanyakan apakah pada bagian tersebut merupakan “vapor” atau “liquid” outlet. Dalam hal ini pilih vapor.
- Kerjakan untuk aliran bagian bawah yang merupakan liquid outlet, sehingga aliran masuk dan keluar dari flash drum sudah lengkap.

**2.5. Menentukan kondisi operasi**

Tahap selanjutnya adalah menentukan kondisi operasi dari aliran masuk.

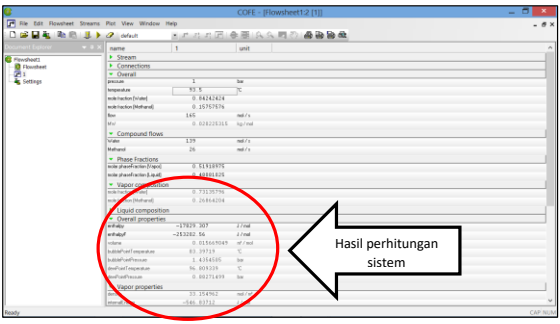
- Letakan kursor pada aliran 1 dan klik mouse kanan → kemudian pilih Edit/view stream (posisi paling bawah)

- Akan muncul jendela, dimana kondisi operasi pada aliran 1 dapat ditentukan, lanjutkan dengan memasukkan kondisi operasi tekanan 1 bar, suhu 93.5$^{o}$C dan kecepatan alir air sebesar 139 mol/det, dan methanol 26 mol/det.
- Apabila satuan tidak sesuai, misalkan tekanan dalam satuan Pa, → klik pada satuan → klik pilih satuan yang sesuai.




Gambar 4.10. Menentukan kondisi operasi pada *Stream*.

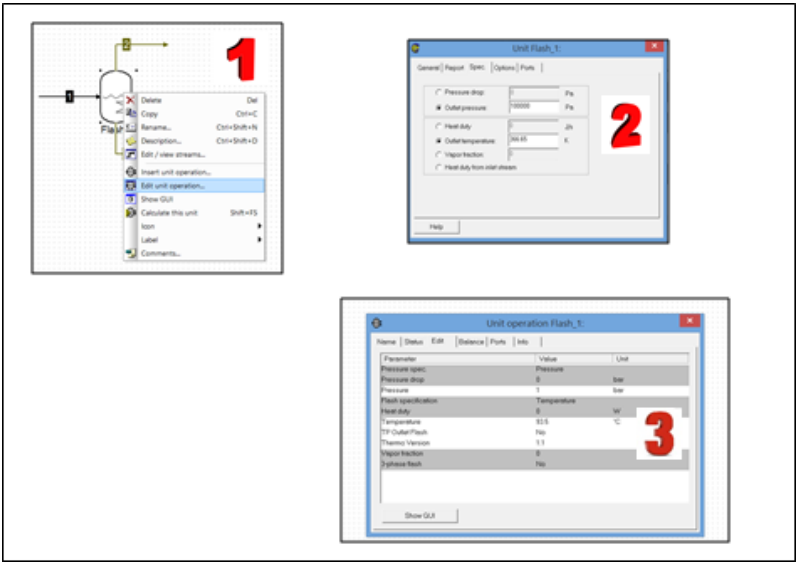
- Setelah semua kondisi operasi di masukkan maka parameter pada *stream* akan dihitung secara otomatis oleh sistem, sebagaimana ditunjukan pada gambar L.11.




Gambar 4.11. Hasil perhitungan pada *Stream*.

Tahap selanjutnya menentukan kondisi spesifik dari alat Unit Operation *flash drum*.

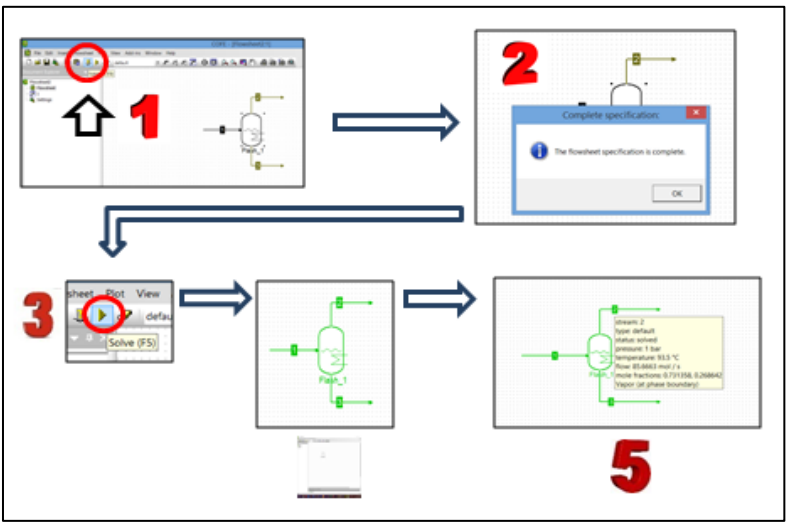
- Klik kanan pada unit (*flash drum*) kemudian pilih Edit Unit Operation
- Pada tab. *Spec* pilih “Pressure drop atau Outlet pressure” dan “Heat duty, Outlet temperature, vapor fraction atau Heat duty from inlet stream” dengan mengklik *radio button*, selanjutnya nilai yang diinginkan diisikan.
- Hasil pilihan akan muncul pada tahap 3, pada bagian yang tidak dipilih, maka warnanya menjadi abu-abu. Tahapan dapat dilihat pada gambar 4.12




Gambar 4.12. Pemilihan kondisi khusus dari Unit Operation

## 2.6. Simulasi

Proses simulasi ini sebaiknya dilakukan pada setiap bagian selesai. Tahapan proses ini dapat dilihat pada gambar 4.13. Tahap satu, adalah memvalidasi flowsheet dengan mengklik tombol atau F8, apakah sudah benar atau belum. Apabila sudah benar akan terlihat komentar "*The flowsheet specification is complete*" sebagaimana ditunjukkan tahap 2 pada gambar 4.13. Apabila keadaan sudah terpenuhi, maka dilanjutkan dengan tahap perhitungan dengan klik button *solve* atau atau F5 sebagaimana ditunjukkan tahap 3. Jika proses perhitungan sudah selesai atau konvergen (apabila ada perhitungan berulang), maka semua aliran dan alat *flash drum* akan semua berubah menjadi warna hijau.



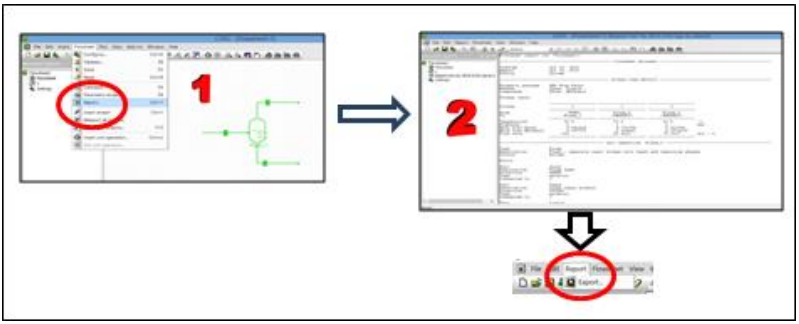

Gambar 4.13. Proses simulasi: validasi flowsheet dan *solve*.

Model sudah dalam kondisi konvergen dan hasil dapat dilihat dengan meletakan kursor mouse pada aliran 2, maka akan muncul sebagai *flag* pada aliran yang ditunjuk (tahap 5) pada gambar 4.13.

## 2.7. Tampilan hasil

Tahap terakhir adalah menampilkan hasil, sebagaimana dapat dilihat pada gambar 4.14 dengan tahapan detail adalah sebagai berikut:

- Untuk mendapatkan hasil perhitungan secara keseluruhan dapat dilakukan dengan memilih flowsheet pada menu bar → pilih report
- Hasil dapat di export: file → export report.




Gambar 4.14. Proses untuk menambahkan hasil simulasi

## 3. ILUSTRASI dengan Reaksi Kimia - REAKTOR

Pada bagian ini akan dilakukan simulasi dengan adanya reaksi kimia, yaitu Ethanol menjadi Diethyl Ether dan $H_2O$, dengan property package yang digunakan adalah Peng-Robinson.

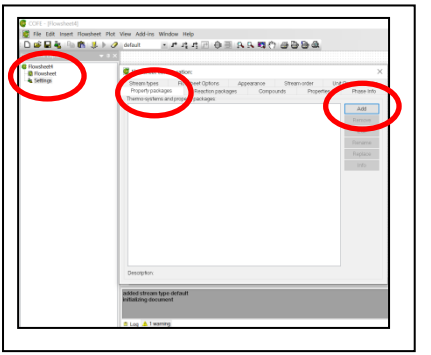
Campuran ethanol dan air ($H_2O$) dengan menggunakan model thermodinamika "Peng-Robinson" dimasukan pada alat Reaktor pada kondisi operasi tekanan 1 atm, suhu 40$^oC$ dengan komposisi 85% mol dan sisanya Air, dengan kecepatan mol sebesar 20 mol/det.
Reaksi yang terjadi adalah: 2 $C_2H_5OH$ → $(C_2H_5)_2O + H_2O$
Reaksi pada kondisi isothermal pada suhu 40$^oC$ dengan

konversi tetap 60% (karena isothermal).
Akan disimulasikan hasil dari reactor berdasarkan neraca massa dan neraca panas dengan menggunakan simulator COCO.

**Tahapan penyelesaian:**

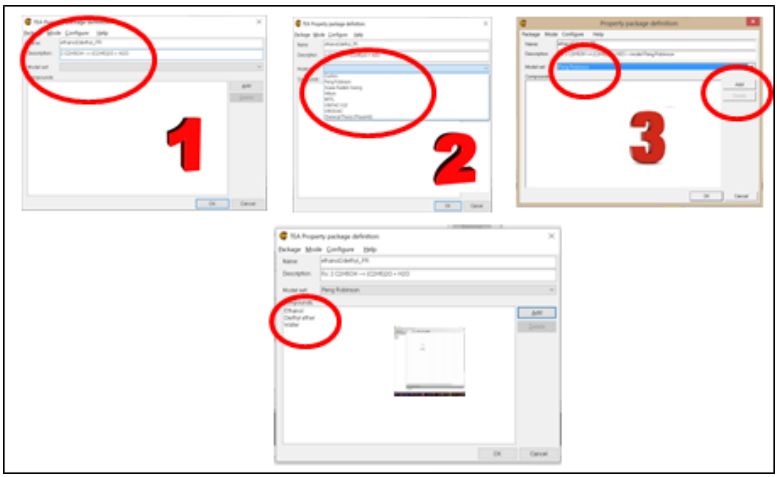
Tahapan sebagaimana pada sub-bab sebelumnya

1. Menentukan Property Package, dengan klik **setting** pada Document Explorer sebelah kiri; kemudian akan muncul window: **Flowsheet configuration**, akan langsung pada tab Property packages, kemudian klik tombol **Add**. Tahapan ini dapat dilihat pada gambar 4.15.

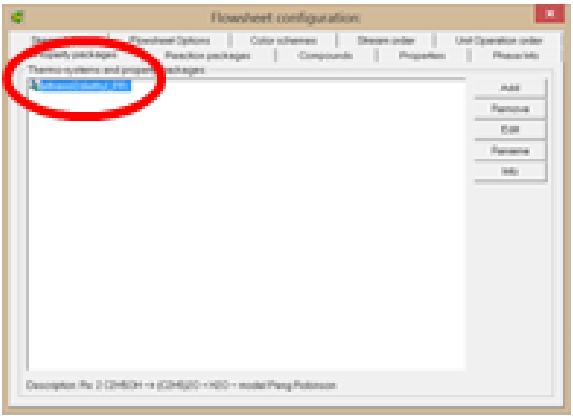


Gambar 4.15. Mengatur konfigurasi sistem dengan tombol **setting**

2. Selanjutnya diketik nama, deskripsi, memilih property packages, dan memilih komponen pada sistem, yaitu “Ethanol, Diethyl Ether dan Air”, tahapan ditunjukkan pada gambar 4.16. Dengan hasil akhir dapat dilihat pada gambar 4.17.



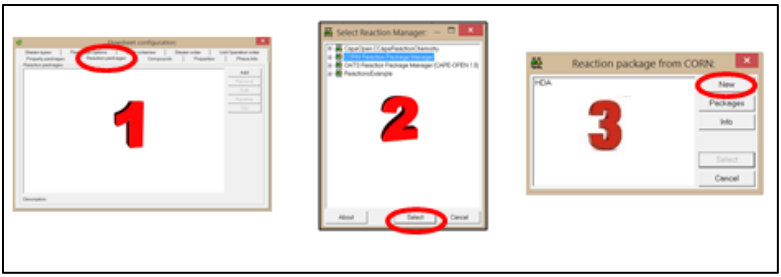

Gambar 4.16. Penambahan Property Packages dan Compounds



Gambar 4.17. Hasil akhir penambahan komponen dan paket properti

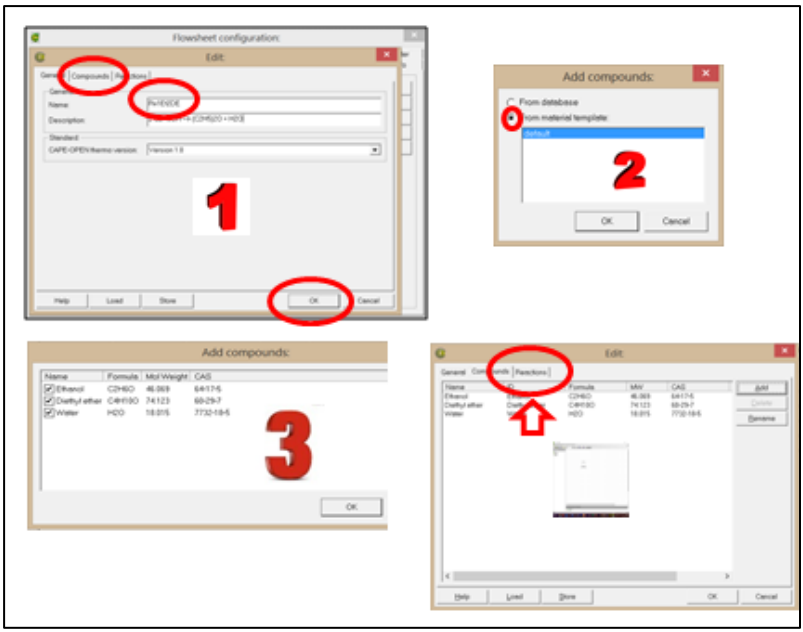
3. Tahap selanjutnya adalah menyusun reaksi pada sistem, dengan jalan membuka jendela *Reaction Package from CORN,* sebagaimana ditunjukkan pada gambar 4.18. Pada jendela *Flowsheet configuration*, pilih tab *Reaction packages*, maka akan muncul jendela *Select Reaction Manager*,

kemudian dipilih paket sistem reaksi yang sesuai, dalam hal ini dipilih *CORN Reaction Package Manager*. Kemudian pilih tombol **New**.




Gambar 4.18. Membuka Jendela *Reaction package*.

4. Pada tahap ini diawali dengan memberikan nama untuk reaksi, karena ada kemungkinan dalam suatu reaksi terdapat reaksi dapat lebih dari satu, sehingga penamaan reaksi hal yang penting, dan untuk mempermudah dalam mengingat dapat diberikan keterangan dalam deskripsi, sebagaimana pada gambar 4.19. Kemudian pilih komponen yang telibat dalam reaksi tersebut, dipilih "*From material template*" karena pada tahap sebelumnya sudah memilih komponen. Kemudian tahap 3 centang komponen pada reaksi ini, maka akan muncul sebagaimana tahap 4.

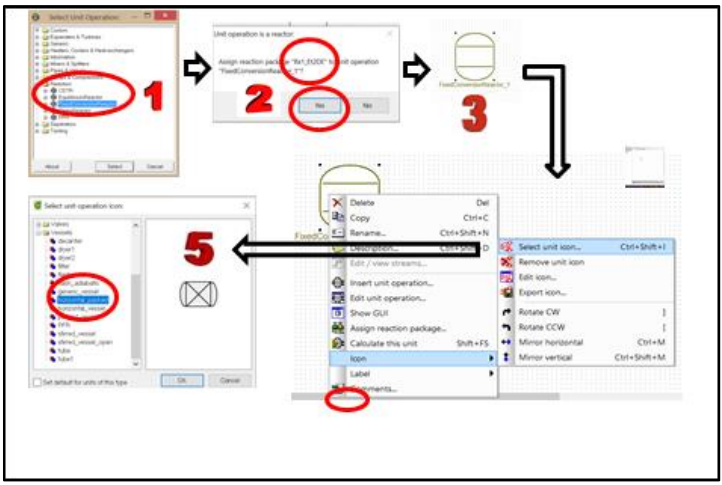



Gambar 4.19. Pemilihan komponen pada tahap Reaksi Kimia

5. Setelah pemilihan komponen, kemudian masuk ke tab "Reaction", setlah itu klik "Create"; dan masukan nama reaksi sebagai ID atau variabel reaksi. Tahap ke 3, isikan koefisien stokhiometri, untuk reaktan diawali dengan minus. Jadi pada reaksi 2 $C_2H_5OH \rightarrow (C_2H_5)_2O + H_2O$, koefisien stokhiometri untuk Ethanol adalah -2, sedangkan yang lain 1. Karena reaksi terjadi pada suhu 40$^oC$, maka komponen reaktan berada pada kondisi cair, sehingga pada bagian phase pilih "liquid" (diantara Vapor, Solid), diakhiri dengan klik Oke, akan muncul sebagaimana tahap 5. Untuk reaksi lebih dari satu, maka tahapan ini dapat diulang.

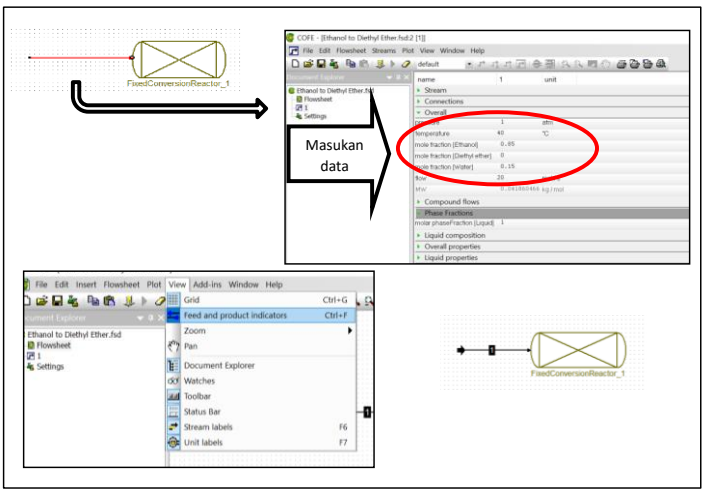

Gambar 4.20. Konfigurasi Reaksi Kimia

6. Setelah konfirgurasi flowsheet selesai, dilanjutkan dengan penambahan alat (Unit Operation), dalam hal ini Reaktor dengan konversi tetap. Pertama klik simbol  , kemudian pada jendela "*Select Unit Operation*" pilih "*Reactors*" dan "*FixedConversionReactor*" sebagaimana pada gambar 4.21 tahap 1. Kemudian akan muncul pertanyaan sebagaimana tahap 2 yang maksudnya Apakah akan menggunakan paket reaksi "Rx1_Et2DE" yang sebelumnya dibuat pada Reaktor yang dipilih? Karena paket tersebut akan digunakan, maka dipilih Yes. Seandainya akan mengganti gambar seperti pada tahap 3, maka klik kanan mouse pada alat UO, kemudian klik "Icon". Selanjutnya akan tampil gambar pada tahap 5, pilih gambar Icon sesuai yang dikehendaki, misalkan dipilih "Vessel", "horizontal_packed".



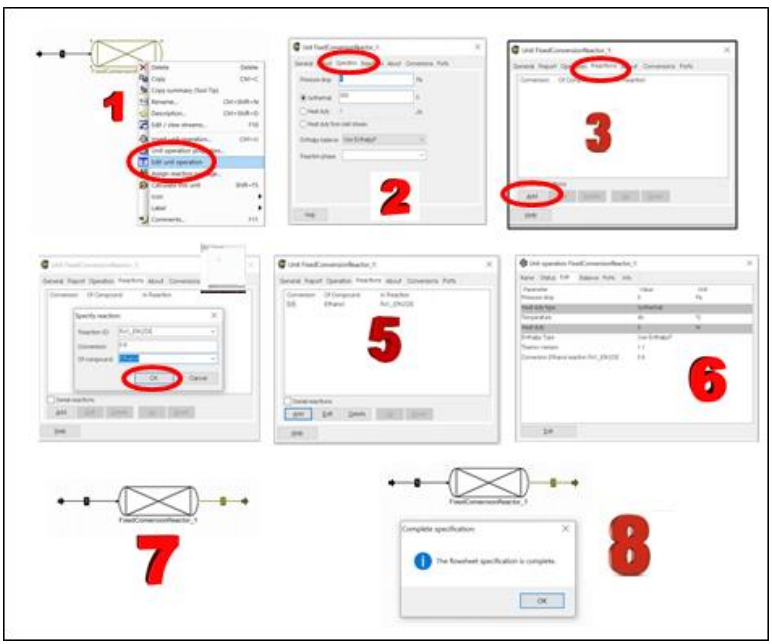

Gambar 4.21. Pemilihan alat dan merubah bentuk tampilan.

7. Tahap selanjutnya tambahkan *stream* (aliran) sebelum masuk reactor. Kondisi operasi sebelum masuk pada reactor: Tekanan 1 atm; Suhu 40$^{o}$C; fraksi mol Ethanol 0.85 dan fraksi mol Air 0.15, serta molar flow masuk 20 mol/sec.



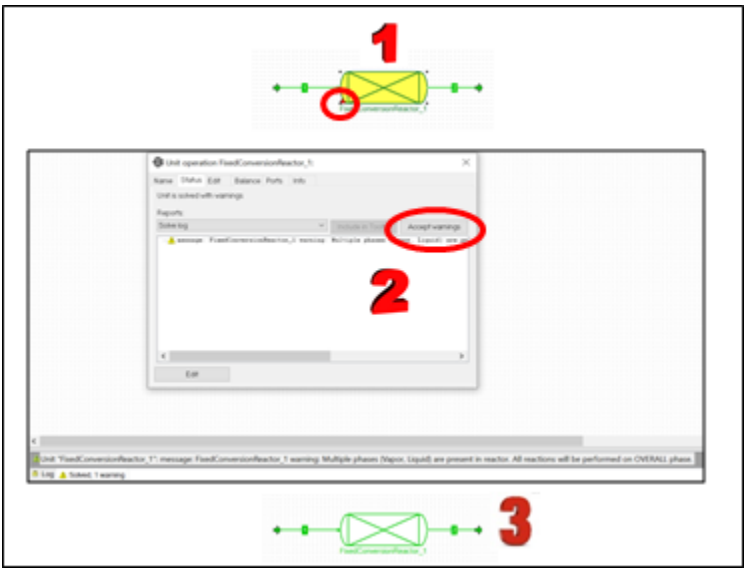

Gambar 4.22. Penambahan Aliran (*Stream*) sebelum masuk reactor.

8. Tahap selanjutnya adalah pengaturan pada Reaktor Konversi tetap, dengan klik kanan pada alat. Pada gambar 4.23, pada tahap (1) *Edit Unit Operation*, tahap (2) pada tab "Operation", diisikan sesuai dengan kondisi operasi. Pada (3) tab "Reaction", klik "Add", maka pada tahap (4) pilih reaksi yang sudah dibuat dengan ID "Rx1_Eth2DE", konversi 0.6 dan sebagai basis perhitungan adalah "Ethanol", klik OK, maka akan tampak pada tahap (5). Apabila reaksi pada reactor lebih dari satu, maka dapat klik Add kembali. Hasilnya pengaturan Reaktor, seperti pada tahap (6) dan (7). Sebelum disimulasikan, dilakukan validasi dari flowsheet dengan klik F8 atau  , maka akan muncul jendela sebagaimana ditunjukkan pada tahap (8).



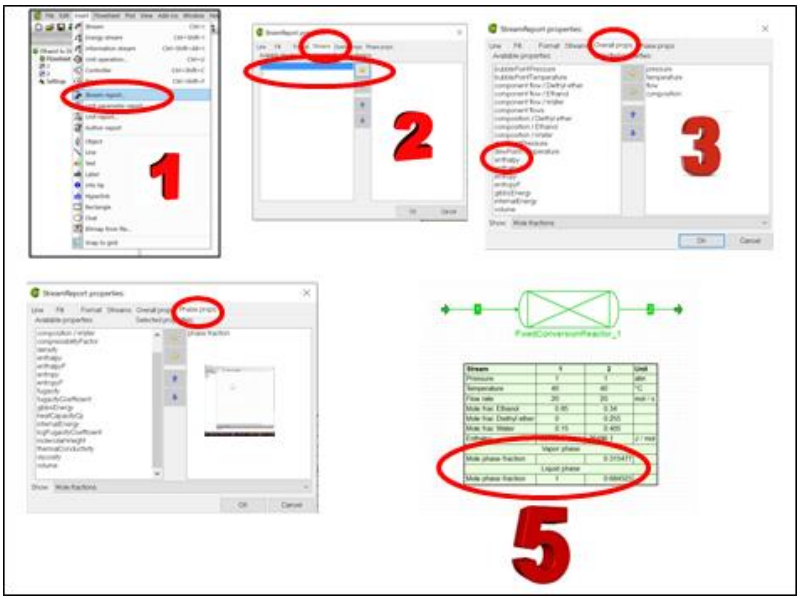

Gambar 4.23.Pengaturan pada Reaktor konversi tetap.

9. Tahap selanjutnya adalah simulasi, dengan klik tombol "Solve" atau F5, maka COCO simulator akan menghitung. Akan tetapi pada hasil simulasi terlihat sebagaimana ditunjukkan pada gambar 4.24, tahap (1) dimana pada alat UO masih terlihat kuning, belum hijau semuanya, dan ada tanda segitiga merah pada bagian kiri bawah, atau ***Warning***. Pada bagian warning (dibawah), klik dua kali, maka komentar akan muncul sebagaimana ditunjukkan pada tahap (2) yaitu *Multiple phases (Vapor, Liquid) are present in Reactor*, artinya pada waktu diset pada Reaktor adalah Liquid, ternyata setelah hasil perhitung terdapat fase Vapor dan Cairan. Oleh karenanya, peringatan (warning) diterima dengan klik tombol "Accept waring", kemudian klik F5 kembali, dan hasilnya simulasi berhasil ditandai semua hijau tahap (3).




Gambar 4.24. Simulasi reactor konversi tetap.

10. Tahap selanjutnya adalah menampilkan hasil simulasi. Pada gambar 4.25 merupakan tahapan untuk memunculkan hasil simulasi pada layar flowchart.

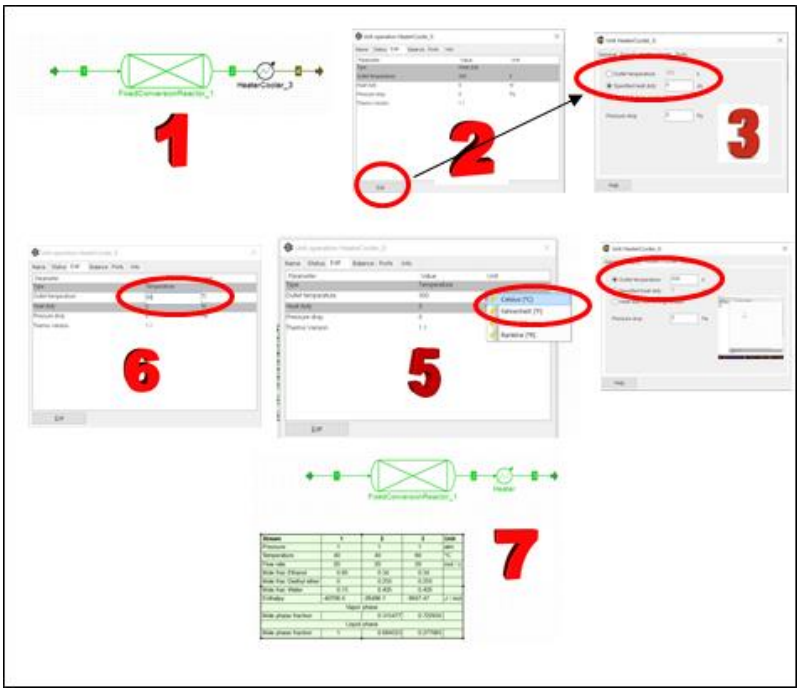



Gambar 4.25. Menampilkan hasil simulasi.

11. Pada gambar 4.25, terlihat bahwa pada saat masuk reactor semua fase cair, dan setelah reaksi kimia terlihat ada dua fase, yaitu fase uap dan fase cair, dengan komposisi fase uap sebesar 31.5% dan sisanya fase cair 68,5%.

Berdasarkan hasil simulasi diatas, komposisi produk reactor untuk Diethyl ether sebesar 25.5% mol.

Apabila simulasi akan dilanjutkan, karena akan memisahkan produk dengan reaktan, dengan menggunakan flash destilasi pada suhu 60$^{o}$C, maka sebelum masuk pada flash perlu ditambah heater.




Gambar 4.26. Simulasi penambahan heater suhu 60$^{o}$C

12. Simulasi penambahan “Heater” setelah Reaktor, tahapannya dapat dilihat pada gambar 4.26 dengan keterangan sebagai berikut:
    1. Penambahan alat dipilih pada kelompok alat “Heater, Cooler & Heat exchangers”, dipilih “HeaterCooler”.
    2. Hubungkan dengan aliran 2, kemudian double klik pada alat, selanjutnya tekan “edit”.
    3. Akan terlihat pada radio button berada pada *Specified heat duty*, hal ini sesuai gambar yang di tahap 2.
    4. Klik *Outlet temperature*, tutup jendela.

5. Terlihat *Outlet temperature* sudah sebagai variabel yang dapat diubah. Karena suhu dalam satuan $^{o}$C, maka klik pada unit suhu, dan dipilih $^{o}$C.
6. Selanjutnya dimasukkan nilai 60.
7. Tahap selanjutnya cek flowrate dan simulasi. Pada simulasi akan keluar **Warning** Kembali pada reactor, perlakukan hal yang pada warning, hal ini karena ada dua fase keluar reactor. Hasil simulasi dapat dilihat dengan menambah *stream report* dengan cara double pada table dan tambahkan.

Hasil simulasi dengan penambahan "Heater" pada suhu 60$^{o}$C terlihat fase uap meningkat menjadi 72.3% dan fase cair 27.7%. Selanjutnya arus 3 akan dimasukkan ke flash untuk disimulasikan.

13. Pada simulasi selanjutkan akan ditambah Flash untuk memisahkan komponen produk yaitu Diethyl Ether dengan reagen yaitu Ethanol dan Air, karena konversinya relative rendah.
14. Penambahan Flash merupakan aliran keluar dari Heater, dan di-rencanakan flash akan beroperasi pada kondisi isothermal suhu 60$^{o}$C. Gambar 4.27 merupakan proses penambahan Flash dari Heater.


Gambar 4.27. Penambahan unit Flash

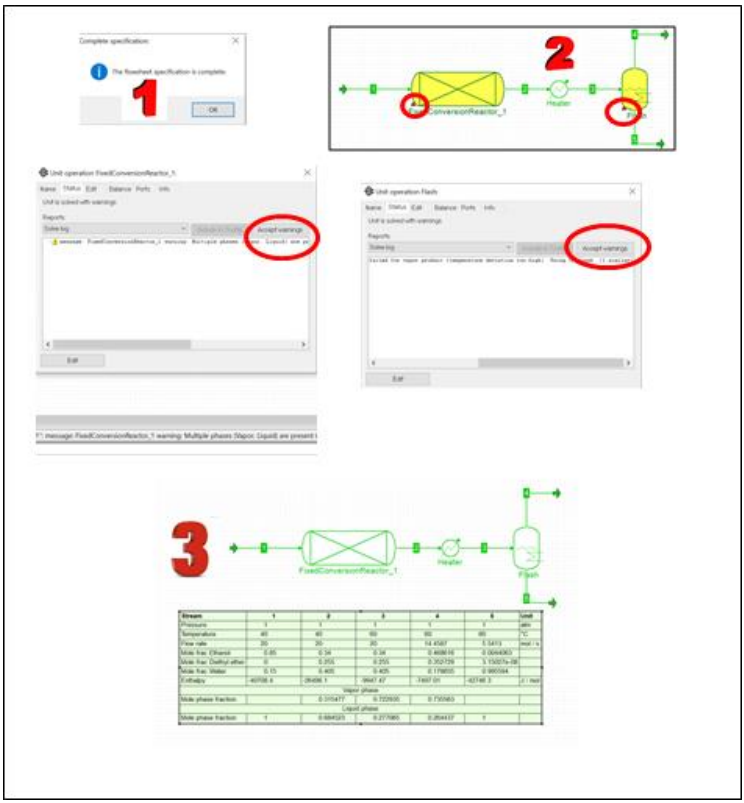
15. Pada gambar 4.27 merupakan penambahan unit Flash setelah unit Heater.
    1. Setelah menambahkan unit Flash, kemudian menyambungkan aliran 3 masuk keunit Flash. Selanjutya ditambah *output* dalam bentuk Uap dan Liquid.
    2. Double klik pada unit Flash, maka parameter yang dapat diatur adalah (yang tidak berwarna abu-abu) *Pressure drop* dan *heat duty*. Padahal diingnkan kondisi isothermal, sehingga parameter yang harus diatur adalah temperature. Untuk merubah, klik edit pada bagian bawah.
    3. Klik *Outlet temperature*
    4. *Outlet temperature* sebagai parameter, tutup jendela.

5. Isikan nilai suhu yang dikehendaki.

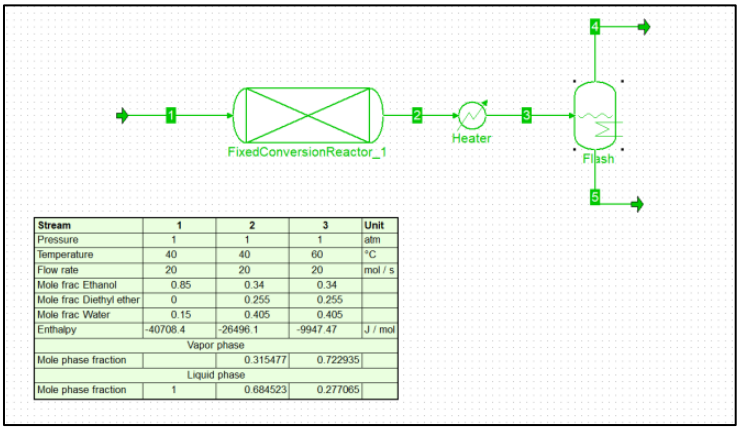
16. Tahap selanjutnya adalah simulasi untuk ketiga unit: Reaktor, Heater dan Flash, dengan tahapan:
    1. Validasi dari Flowsheet, apabila tidak ada masalah akan muncul sebagaimana pada gambar 4.28, kemudian dilakukan simulasi dengan F5.
    2. Hasil simulasi masih belum hijau semua, dan ada Warning dari Unit, pada bagian bawah diklik akan muncul jendela yang menunjukkan permasalahan:
        i. pada reactor kondisi hasil Uap dan Cairan, permasalahan ini muncul karena *Solve* diulang dari awal, klik pada *Accept warnings*.
        ii. Pada flash juga muncul, penyelesaiannya Using TP Flash, klik pada *Accept warnings*.

17. Kemudian dilakukan simulasi ulang F5, dan semua unit berhasil dieksekusi. Kemudian stream report diperbaiki dengan menambah aliran-aliran yang belum ada.

Berdasarkan hasil simulasi terlihat pada arus ke 5, produk flash sudah hampir tidak ada Diethyl ether $3.15 \times 10^{-8}$. Sehingga apabila diperlukan pada produk Uap flash dapat direcycle untuk mengurang kebutuhan bahan baku. Akan tetapi perlu diteliti ulang mengenai masalah ekonomi.




Gambar 4.28. Simulasi Reaktor, Heater dan Flash.




| Stream | 1 | 2 | 3 | Unit |
|---|---|---|---|---|
| Pressure | 1 | 1 | 1 | atm |
| Temperature | 40 | 40 | 60 | °C |
| Flow rate | 20 | 20 | 20 | mol / s |
| Mole frac Ethanol | 0.85 | 0.34 | 0.34 | |
| Mole frac Diethyl ether | 0 | 0.255 | 0.255 | |
| Mole frac Water | 0.15 | 0.405 | 0.405 | |
| Enthalpy | -40708.4 | -26496.1 | -9947.47 | J / mol |
| Vapor phase | | | | |
| Mole phase fraction | | 0.315477 | 0.722935 | |
| Liquid phase | | | | |
| Mole phase fraction | 1 | 0.684523 | 0.277065 | |

# MODELING dan SIMULASI
# PADA PROSES INDUSTRI KIMIA

Buku ini ditulis berdasarkan metoda dan hasil penelitian teknologi informasi khususnya dalam bidang Teknik Kimia atau Komputasi Proses. Akan tetapi, buku ini ditulis bukan hanya pada peningkatan kemampuan untuk mengetahui dan memahami *knowledge* saja, akan tetapi sampai pada tingkat kemampuan untuk pengembangan **mengaplikasi *skill*** sistem komputasi proses, dengan syarat pengguna harus praktek. Untuk mempraktekkan bahan yang ada dalam buku ini diperlukan perangkat lunak. Oleh karenanya yang unik dari buku ini adalah semua menggunakan perangkat lunak yang bebas bayar, bebas pengembangan atau lebih dikenal dengan *Free Open Source Software* (***FOSS***). Dengan demikian, keuntungan dengan membaca, mempraktekkan dari buku ini bersifat murah dan legal.

- Bab I dari buku ini membahas mengenai pengembangan penelitian pemodelan sistem komputasi proses dengan menggunakan perangkat lunak **Scilab** yang serupa dengan perangkat lunak Matlab, akan tetapi bebas bayar. Scilab maupun Matlab merupakan perangkat lunak untuk pengembangan pemodelan matematis.
- Bab II membahas mengenai penggabungan antara perangkat lunak *FOSS* dengan **Scilab** khususnya dalam hal penggabungan paket Thermodinamika pada *FOSS* khususnya pada **CAPE-OPEN simulator** (***Computer Aided Process Engineering***).
- Bab III membahas penggabungan Scilab pada perangkat lunak CAPE-OPEN simulator, sehingga untuk Unit Operation atau alat yang bersifat khusus agar dapat dikombinasikan pada simulasi sistem proses.
- Bab IV membahas secara detail akan tetapi bersifat simple untuk perangkat lunak **COCO** (**CAPE OPEN to CAPE OPEN**), sehingga pengguna dapat mahir menggunakan perangkat lunak tersebut dengan syarat harus dipraktekkan.

Sebuah harapan, pengembangan ilmu Sistem Komputasi Proses dalam bidang tridarma perguruan tinggi: pendidikan, penelitian dan pengabdian kepada masyarakat dapat berkembang lebih pesat dalam era digitalisasi serta konektivitas (*networking*) – IoT (***Internet of Thing***) khususnya pada Universitas Diponegoro.

diterbitkan oleh :
UNDIP PRESS
SEMARANG

ISBN 978-979-0977-54-9
9 789790 977549