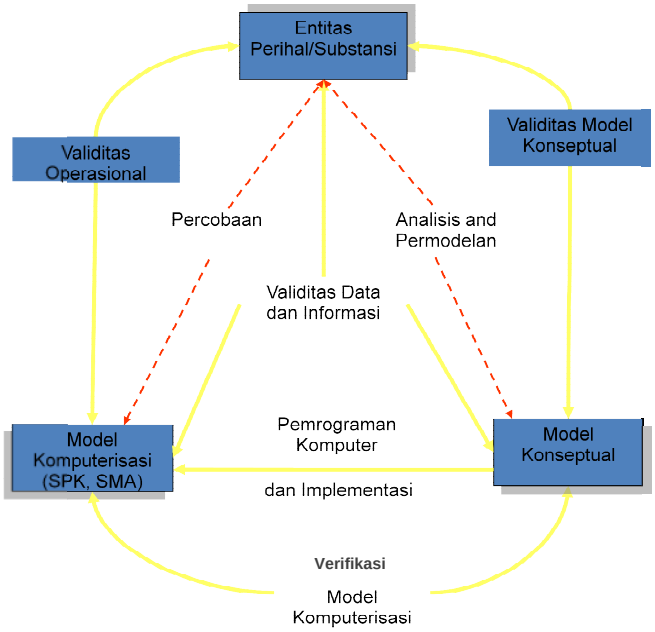
VERIFIKASI DAN VALIDASI DALAM SIMULASI MODEL

Andi Hasad
andihasad@yahoo.com
Sekolah Pascasarjana IPB, Departemen Ilmu Komputer
Jl. Raya Darmaga, Kampus IPB Darmaga, Wing 20 Level 5-6,
Bogor - Jawa Barat, Indonesia,16680
Telp. 0251 8625584, Fax. 0251 8625584

Verifikasi bertujuan untuk membuktikan bahwa sesuatu ada atau benar, atau untuk memastikan bahwa sesuatu adalah benar (*Verify : to prove that something exists or is true, or to make certain that something is correct*), sedangkan validasi bertujuan untuk membuat sesuatu yang resmi diterima atau disetujui, terutama setelah memeriksanya (*Validate : to make something officially acceptable or approved, especially after examining it*) (Cambridge Dictionaries Online, 2011).

Menurut Sargent (1999) verifikasi model yang terkomputerisasi (*computerized*) memastikan bahwa pemrograman komputer dan implementasi model konseptual adalah benar. Untuk membantu memastikan bahwa sebuah program komputer adalah benar, desain program dan prosedur pengembangan pada bidang *software engineering* sebaiknya digunakan dalam pengembangan dan implementasi program komputer. Hal ini termasuk desain *object oriented*, desain *top-down*, pemrograman terstruktur dan *modularity* program. Sedangkan validasi model terkomputerisasi (*computerized*) berarti memastikan bahwa program komputer dari model yang terkomputerisasi beserta implementasinya adalah valid (sah dan diterima) atau tidak valid.

Dalam proses validasi, terdapat 3 pendekatan utama yang digunakan untuk menentukan kapan sebuah model simulasi adalah valid atau tidak valid. Setiap pendekatan membutuhkan tim pengembangan model untuk melakukan verifikasi dan validasi sebagai bagian dari proses pengembangan model. Pendekatan umum yang paling banyak untuk tim pengembangan adalah membuat keputusan bahwa model tersebut valid. Ini adalah keputusan subyektif yang berdasarkan pada berbagai pengujian dan evaluasi yang dilakukan sebagai bagian dari proses pengembangan model.



Gambar 1 Proses Pemodelan Sistem (Sargent, 1999)

Pendekatan yang lain, sering disebut verifikasi dan validasi independen, digunakan oleh pihak ketiga (independen) untuk menentukan kapan model itu valid. Pihak ketiga adalah independen dari kedua sisi, baik dari sisi tim pengembangan model maupun sponsor/pengguna model. Setelah sebuah model dikembangkan, pihak ketiga melakukan evaluasi untuk menentukan validitasnya. Berdasarkan validasi ini, pihak ketiga membuat sebuah keputusan subyektif pada validitas dari model tersebut.

Tabel 1 Evaluasi Untuk Validitas Model Konseptual

Category/Item	Technique(s) Used	Justification for Technique Used	Reference to Supporting Report	Result/ Conclusion	Confidence In Result
• Theories • Assumptions • Model representation	• Face validity • Historical • Accepted approach • Derived from empirical data • Theoretical derivation				

Strengths	
Weaknesses	

Overall evaluation for Computer Model Verification	Overall Conclusion	Justification for Conclusion	Confidence In Conclusion

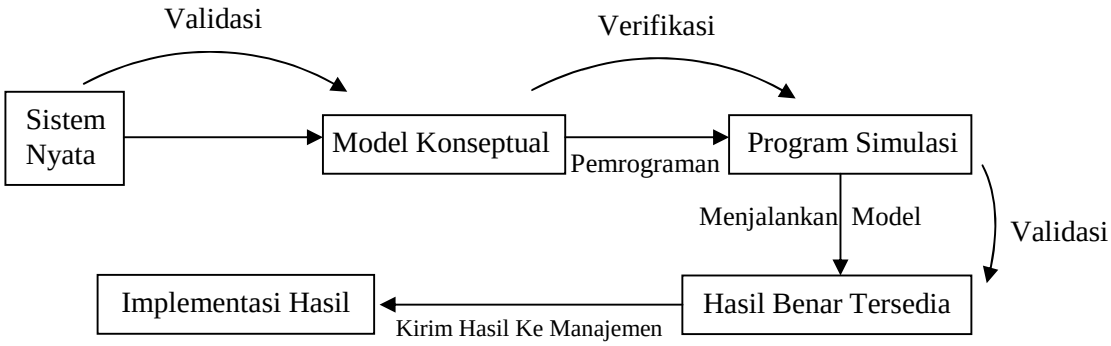
Pendekatan terakhir untuk menentukan kapan sebuah model adalah valid untuk digunakan adalah model penentuan nilai (*scoring*). Skor atau bobot

ditentukan secara subyektif ketika melakukan proses validasi dari berbagai aspek dan kemudian dikombinasikan untuk menentukan nilai (skor) kategori dan skor keseluruhan untuk model simulasi. Sebuah model simulasi dinyatakan valid jika skor kategori dan skor keseluruhan lebih besar dari beberapa skor lainnya. Pendekatan ini tidak sering digunakan dalam praktek.

Menurut Hoover dan Perry (1989) verifikasi adalah pemeriksaan apakah program komputer simulasi berjalan sesuai dengan yang diinginkan, dengan pemeriksaan program komputer. Verifikasi memeriksa penerjemahan model simulasi konseptual (diagram alur dan asumsi) ke dalam bahasa pemrograman secara benar (Law dan Kelton, 1991), sedangkan validasi adalah proses penentuan apakah model, sebagai konseptualisasi atau abstraksi, merupakan representasi berarti dan akurat dari sistem nyata ? (Hoover dan Perry, 1989). Validasi adalah penentuan apakah model konseptual simulasi (sebagai tandingan program komputer) adalah representasi akurat dari sistem nyata yang sedang dimodelkan (Law dan Kelton, 1991).

- **Verifikasi dan Validasi dalam Simulasi**

  Ketika membangun model simulasi sistem nyata, kita harus melewati beberapa tahapan atau level pemodelan. Seperti yang dapat dilihat pada Gambar 1, pertama kita harus membangun model konseptual yang memuat elemen sistem nyata.



Gambar 2 Relasi verifikasi, validasi dan pembentukan model kredibel

Dari model konseptual ini kita membangun model logika yang memuat relasi logis antara elemen sistem juga variabel *eksogenus* yang

mempengaruhi sistem. Model kedua ini sering disebut sebagai model diagram alur. Menggunakan model diagram alur ini, lalu dikembangkan program komputer, yang disebut juga sebagai model simulasi, yang akan mengeksekusi model diagram alur.

Pengembangan model simulasi merupakan proses iteratif dengan beberapa perubahan kecil pada setiap tahap. Dasar iterasi antara model yang berbeda adalah kesuksesan atau kegagalan ketika verifikasi dan validasi setiap model. Ketika validasi model dilakukan, kita mengembangkan representasi kredibel sistem nyata, ketika verifikasi dilakukan, kita memeriksa apakah logika model diimplementasikan dengan benar atau tidak. Karena verifikasi dan validasi berbeda, teknik yang digunakan untuk yang satu tidak selalu bermanfaat untuk yang lain. Baik untuk verifikasi atau validasi model, kita harus membangun sekumpulan kriteria untuk menilai apakah diagram alur model dan logika internal adalah benar dan apakah model konseptual representasi valid dari sistem nyata. Bersamaan dengan kriteria evaluasi model, kita harus spesifikasikan siapa yang akan mengaplikasikan kriteria dan menilai seberapa dekat kriteria itu memenuhi apa yang sebenarnya.

Tabel 1 Perbandingan verifikasi dengan validasi dalam berbagai model

Model	Verifikasi	Validasi
Konseptual		Apakah model mengandung semua elemen, kejadian dan relasi yang sesuai?
		Apakah model dapat menjawab pertanyaan pemodelan?
Logika	Apakah kejadian direpresentasikan dengan benar?	Apakah mode memuat semua kejadian yang ada pada model konseptual?
	Apakah rumus matematika dan relasi benar?	
	Apakah ukuran statistik dirumuskan dengan benar?	Apakah model memuat semua relasi yang ada dalam model konseptual?
Komputer atau simulasi	Apakah kode komputer memuat semua asapek mode logika?	Apakah model komputer merupakan representasi valid dari sistem nyata?
	Apakah statistik dan rumus dihitung dengan benar?	Dapatkah model komputer menduplikasi kinerja sistem nyata?
	Apakah mode mengandung kesalahan pengkodean?	Apakah output model komputer mempunyai kredibilitas dengan ahli sistem dan pembuat keputusan?

Praktisi simulasi harus dapat menentukan aspek apa saja, dari sistem yang kompleks, yang perlu disertakan dalam model simulasi.

Petunjuk umum dalam menentukan tingkat kedetailan yang diperlukan dalam model simulasi :

a. Hati-hati dalam mendefinisikan
b. Model-model tidak valid secara universal
c. Memanfaatkan ‘pakar’ dan analisis sensitivitas untuk membantu menentukan level detil model

- **Validasi Model Konseptual**

Validasi model konseptual adalah proses pembentukan abstraksi relevan sistem nyata terhadap pertanyaan model simulasi yang diharapkan akan dijawab. Validasi model simulasi dapat dibayangkan sebagai proses pengikat dimana analis simulasi, pengambil keputusan dan manajer sistem setuju aspek mana dari sistem nyata yang akan dimasukkan dalam model, dan informasi apa (output) yang diharapkan akan dihasilkan dari model. Tidak ada metode standar untuk validasi model konseptual, kita hanya akan melihat beberapa metode yang berguna untuk validasi.

- Representasi Kejadian Sistem

Metode ini menggunakan graf kejadian seperti yang digunakan dalam pengembangan model simulasi. Teknik pembuatan grafnya juga sama. Kita harus mendefinisikan dengan jelas relasi kondisional antar kejadian. Representasi graf dapat digunakan sebagai jembatan ke model logis (model diagram alur) juga sebagai alat bantu komunikasi antara analis simulasi, pengambil keputusan dan manajer. Hampir sama dengan graf kejadian adalah model diagram alur, merepresentasikan aliran entitas melalui sistem.

Secara konseptual, kita modelkan sistem sebagai interaksi kejadian:

a. Pemakai melakukan koneksi ke sistem
b. Pemakai terhubung dan sesi mulai
c. Pemakai menyudahi sesi
d. Pemakai yang ditolak mencoba koneksi ke sistem

- Identifikasi Eksplisit Elemen yang Harus Ada dalam Model

Pada umumnya model konseptual tidak dapat memasukkan semua detil sistem nyata, melainkan hanya elemen yang relevan dengan pertanyaan yang diharapkan akan dijawab. Dalam pembuatan model konseptual, semua kejadian, fasilitas, peralatan, aturan operasi, variabel status, variabel keputusan dan ukuran kinerja harus jelas diidentifikasikan dan akan menjadi bagian dari model simulasi. Kita juga harus mengidentifikasikan dengan jelas semua elemen yang tidak akan dimasukkan dalam model simulasi. Analis simulasi, pengambil keputusan dan manajer harus bergabung untuk memutuskan berapa banyak sistem nyata harus dimasukkan untuk menghasilkan representasi valid sistem nyata.

Filosofi yang digunakan untuk memutuskan berapa banyak sistem nyata harus dimasukkan dalam model simulasi adalah :

1. Memasukkan semua aspek sistem yang dapat mempengaruhi perilaku sistem dan menyederhanakan model begitu dapat memahami elemen relevan sistem.
2. Memulai dengan model sederhana sistem dan biarkan model berkembang semakin kompleks sejalan degan semakin jelasnya eleme-elemen sistem yang harus dimasukkan dalam model untuk menjawab pertanyaan.
3. Mengeluarkan usaha dan waktu yang lebih banyak dengan mereka yang lebih memahami sistem nyata, identifikasikan semua elemen yang akan memberikan dampak signifikan akan jawaban pertanyaan model yang diharapkan akan dijawab.

Sistem komputer *time-shared* sebagai berikut:

- Kejadian :
  1. Pemakai berusaha koneksi ke sistem
  2. Pemakai terhubung dan sesi mulai
  3. Pemakai menyudahi sesi
- Fasilitas :
  1. Komputer server
  2. Port
- Variabel status :
  1. Jumlah port yang sedang digunakan

2. Waktu pemanggilan berikutnya
3. Waktu akhir koneksi port ke-i
4. Mengindikasikan apakah port sibuk atau menganggur

- Ukuran kinerja:
  1. Waktu kumulatif pemakai terhubung ke sistem
  2. Jumlah total pemakai memanggil sistem
  3. Jumlah total panggilan yang terhubung
  4. Jumlah total panggilan yang gagal terhubung
  5. Utilitas port
- Variabel keputusan:
  1. Jumlah port
  2. Ekspektasi lama sesi pemakai
- Aturan operasional :
  1. Klien mencoba berulang-ulang sampai tersambung.
- Aspek sistem nyata yang tidak dimasukkan diantaranya:
  1. Klien tidak akan mencoba hubungan lagi pada periode waktu tertentu jika menemukan port semua sibuk.
  2. Kerusakan fasilitas

- **Verifikasi dan Validasi Model Logis**

  Bentuk model logis tergantung dari bahasa pemrograman yang akan digunakan. Jika model konseptual sudah dibangun dengan baik, verifikasi model konseptual bukan pekerjaan kompleks. Ada beberapa pertanyaan yang harus dijawab sebelum kita yakin bahwa model logis merepresentasikan model konseptual. Salah satu pendekatan yang digunakan untuk verifikasi model logis adalah dengan fokus pada :

  1. Apakah kejadian dalam model diproses dengan benar ?
  2. Apakah rumus matematika dan relasi dalam model valid ?
  3. Apakah statistik dan ukuran kinerja diukur dengan benar ?.

- Verifikasi dan Validasi Pemrosesan Kejadian

  a. Validasi bahwa model logis mengandung semua kejadian dalam model konseptual

b. Verifikasi hubungan di antara kejadian
c. Verifikasi bahwa model logis memproses kejadian secara simultan dengan urutan benar.
d. Verifikasi bahwa semua variabel status yang berubah karena terjadinya suatu kejadian diperbaiki dengan benar.

Metode umum yang digunakan untuk verifikasi dan validasi pemrosesan kejadian dalam model logis adalah *structured walk-through*, dimana pengembang model logis harus menjelaskan (walk through) logika detil model ke anggota lain tim pengembang model simulasi.

Kembali ke kasus sistem komputer time-shared, verifikasi dan validasi pemrosesan kejadian bisa diperiksa mulai dari kondisi T≥T_FINAL? sampai dengan N=K?.

- Verifikasi Rumus dan Relasi

  Termasuk dalam model simulasi adalah sejumlah eksplisit atau implisit fungsi dan relasi matematik. Penurunan angka acak dan variabel acak berbasis matematik, dan dalam model simulasi pada umumnya ada hukum konservasi yang harus dipenuhi. Untuk kasus sistem komputer time-shared, periksa kembali rumus dan relasi yang didefinisikan pada mode logika berikut :

```
N_CALLS=N_CALLS+1;
CUM_CONNECT_TIME=CUM_CONNECT_TIME+(T_NEXT_CALL_T)*N;
T=T_NEXT-CALL;
T_NEXT_CALL=T+F_NEXT_CALL;
CUM_CONNECT_TIME=CUM_CONNECT_TIME+(T_CALL_END(i)-T)*N;
N=N-1;T=T_CALL-END(i);
set PORT_STATUS (i) menganggur
N=N+1;
cari port yang menganggur (i);
T_CALL_END(i)=T+F_CONNECT_TIME;CUM_CONNECT=CUM_CONNECT+1
```

- Verifikasi Statistik dan Ukuran Kinerja

  Kesalahan umum yang terjadi dalam pemodelan simulasi adalah gagal memperbaharui statistik relevan dan ukuran kinerja secara tepat ketika suatu kejadian terjadi. Salah satu cara yang dapat digunakan untuk verifikasi bahwa statistik dan ukuran kinerja diperbaharui dengan benar adalah menggunakan graf kejadian. Dalam kebanyakan bahasa simulasi, beberapa

tipe ukuran statistik dapat dikumpulkan secara otomatis saat simulasi dieksekusi. Oleh karena itu, ukuran statistik dibangun dalam metode yang transparan ke analis, sehingga mengurangi kesempatan kesalahan statistik.

Ketika model logis dibangun, adalah penting melakukan validasi bahwa sttaistik dan ukuran kinerja adalah satu-satunya yang perlu dijawab.

- **Verifikasi Model Komputer**

  Teknik Verifikasi Progam :

  Teknik 1. Membuat dan debug program komputer dalam modul-modul atau subprogram-subprogram

  Teknik 2. Membuat program komputer secara bersama-sama (lebih dari satu orang)

  Teknik 3. Menjalankan simulasi dengan berbagai variasi parameter input dan memeriksa apakah outputnya reasonable

  Teknik 4. Melakukan “trace”. Teknik ini merupakan salah satu teknik yang powerful yang dapat digunakan untuk mendebug program simulasi event diskrit.

  Teknik 5. Model sebaiknya dapat dijalankan (jika memugkinkan) dengan asumsi sederhana.

  Teknik 6. Untuk beberapa model simulasi, akan lebih bermanfaat untuk melakukan observasi sebuah animasi dari output simulasi.

  Teknik 7. Menuliskan mean sampel dan variasi sampel untuk setiap probabilitas distribusi input simulasi, dan bandingkan dengan mean dan variansi yang diinginkan (misalnya secara historis)

  Teknik 8. Menggunakan paket simulasi

  Model komputer diverifikasi dengan menunjukkan bahwa program komputer adalah implementasi tepat model logis. Beberapa metode yang digunakan untuk verifikasi model komputer adalah unik terhadap simulasi, sementara metode verifikasi lain sama dengan yang digunakan dalam setiap pengembangan perangkat lunak lainnya. Verifikasi model komputer sangat tergantung dengan bahasa pemrograman yang digunakan dan tidak ada metodologi umum yang disetujui. Verifikasi model komputer sering

membutuhkan imaginasi dan keahlian tinggi analis, dan ini adalah satu aktivitas dalam proyek simulasi yang dilakukan tanpa bantuan pengambil keputusan dan manajer.

Verifikasi model komputer dapat dilakukan dengan :

- Metode pemrograman terstruktur
- Penelusuran model simulasi
- Pengujian
- Pengujian relasi logis
- Verifikasi dengan model analitis
- Verifikasi menggunakan grafik

- Metode Pemrograman Terstruktur

Prinsip pemrograman terstruktur termasuk :

1. Desain Atas-Bawah (Top-Down). Program dirancang mulai dari proses level tertinggi yang kemudian didekomposisi menjadi modul pendukung yang kemudian dapat didekomposisi lagi.
2. Modularitas : setiap modul pendukung bertanggung jawab untuk satu fungsi.
3. Perbaikan langkah demi langkah : setiap modul dikembangkan dengan perbaikan langkah-demi-langkah dan diakhiri dengan kode khusus-bahasa pemrograman. Beberapa langkah perbaikan sudah terjadi pada pengembangan model logis.
4. Pemampatan modul: modul harus pendek.
5. Kontrol terstruktur : semua kode kontrol harus sangat terstruktur menggunakan pernyataan IF-THEN-ELSE, WHILE, REPEAT-UNTIL, FOR DAN CASE. Penggunan pernyataan GOTO harus dihindarkan.

- Penelusuran Simulasi

Beberapa bahasa simulasi menyediakan kemampuan-terpasang penelusuran simulasi sebagaimana terjadinya. Ketika model simulasi diprogram menggunakan bahasa umum (seperti FORTRAN, Pascal, C++), tentu saja analis harus membangun kemampuan penelusuran dalam kode program. Ketika membangun memprogram model logika, mekanisme

penelusuran simulasi harus dimasukkan sebagai bagian dari disain program dan tidak ditutupi ketika ada kesalahan dalam program komputer.

- Pengujian

  Dua pendekatan pengujian adalah *bottom-up* dan *top-down*. Pada pendekatan *bottom-up,* yang terendah, modul dasar pada umumnya diuji dan diverifikasi terlebih dahulu. Pendekatan kadang-kadang disebut dengan pengujian unit. Setelah modul dasar diuji, uji terintegrasi dilakukan dimana *interface* diantara kedua modul diuji. Pendekatan *bottom-up* ini berlanjut terus sampai model dapat diuji sebagai sistem tunggal. Bagian terpenting dalam pengujian adalah seleksi data uji. Keuntungan pengujian modul paling rendah terlebih dahulu adalah pengujian itu membutuhkan himpunan data uji yang lebih kecil daripada modul integrasi yang lebih besar. Modul dapat diuji menggunakan driver yang menurunkan data uji, dan kemudian modul dieksekusi.

  Pada pendekatan *top-down*, pengujian dimulai dengan modul utama dan secara inkremntal bergerak turun ke modul paling rendah. Dalam pengujian *top-down*, rutin (*routine*) *dummy* dibutuhkan untuk mensimulasikan fungsi modul level paling rendah. Keuntungan pendekatan *top-down* adalah proses berlangsung secara logika, paralel dengan aliran program. Programmer dan manajer biasanya lebih menyukai pendekatan top-down karena keberlangsungna proses dapat dilihat. Setelah model diuji baik dengan pendekatan *bottom-up* ataupun *top-down*, model harus diuji coba dengan kondisi paling ekstrim. Jika dipilih dengan hati-hati, hasil simulasi dengan kondisi ekstrim dapat diprediksi.

- Pengujian Relasi Logis

  Relasi ini dapat didasarkan pada hukum konservasi atau secara statistik. Jika relasi ini tidak diperhatikan, maka program bukan implementasi benar dari model logis. Titik paling sesuai untuk memeriksa relasi itu adalah ketika model berjalan tahap demi tahap. Secara tipikal, kesalahan pemrograman tidak acak dan berdistribusi secara uniform, tetapi berkumpul secara kluster.

- **Validasi Model Simulasi**

Persfektif Umum Simulasi:

1. Eksperimen dengan model simulasi untuk eksperimen sistem aktual
2. Kemudahan atau kesulitan dari proses validasi tergantung pada kompleksitas sistem yang dimodelkan
3. Sebuah model simulasi dari sebuah sistem yang kompleks hanya dapat menjadi pendekatan terhadap aktual sistem
4. Sebuah model simulasi sebaiknya selalu dibangun untuk sekumpulan tujuan tertentu
5. Sebuah buku catatan dari asumsi-asumsi model simulasi sebaiknya diupdate berkala
6. Sebuah model simulasi sebaiknya divalidasi relatif terhadap ukuran kinerja yang akan digunakan untuk pengambilan keputusan
7. Pembentukan model dan validasi sebaiknya dilakukan sepanjang pensimulasian
8. Pada umumnya tidak mungkin untuk membentuk validasi statistik secara formal diantara data output model dengan data output sistem aktual

Langkah 1. Membangun sebuah model dengan usaha melibatkan informasi semaksimal mungkin :

- Berdiskusi dengan para ‘pakar’ sistem
- Melakukan observasi terhadap sistem
- Memanfaatkan Teori yang ada
- Memanfaatkan hasil dari Model simulasi yang sama dan relevan
- Menggunakan pengalaman atau intuisi
- Memanfaatkan Teori yang ada
- Memanfaatkan hasil dari Model simulasi yang sama dan relevan
- Menggunakan pengalaman atau intuisi

Langkah 2. Menguji asumsi-asumsi model secara empiris.

Jika distribusi probabilitas secara teoritis cocok dengan observasi dan digunakan sebagai input untuk model simulasi,

dapat diuji dengan pembuatan grafik dan uji *goodness-of-fit*. Jika beberapa himpunan data diobservasi untuk fenomena *random* yang sama, maka perbaikan dari penggabungan data tersebut dapat ditentukan dengan uji Kruskal-Wallis Salah satu utiliti yang sangat berguna adalah analisis sensitivitas

Langkah 3. Menentukan seberapa representatif output simulasi.

Prosedur Statistik untuk membandingkan data output dari observasi dunia nyata dan simulasi :

- Korelasi pendekatan inspeksi
- Pendekatan pendugaan selang kepercayaan berdasarkan data independen
- Pendekatan *Time Series*

Validasi model simulasi dilakukan dengan partisipasi analis, pengambil keputusan manajer sistem. Uji validasi model adalah apakah pengambil keputusan dapat mempercayai model yang digunakan sebagai bagian dari proses pengambilan keputusan.

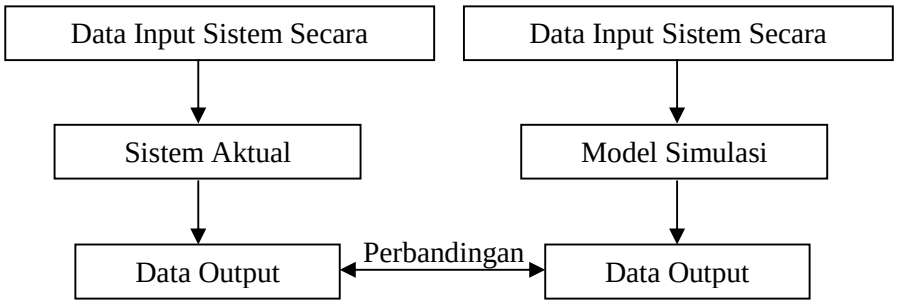
Tidak ada teknik tunggal untuk melakukan validasi model. Prosedur validasi model simulasi tergantung dari sistem yang sedang dimodelkan dan lingkungan pemodelan. Beberapa metode validasi adalah:

1. Perbandingan output simulasi dengan sistem nyata
2. Metode Delphi
3. Pengujian Turing
4. Perilaku ekstrim.

- **Perbandingan Output Simulasi dengan Sistem Nyata**

Membandingkan output ukuran kinerja model simulasi dengan ukuran kinerja yang sesuai dari sistem nyata adalah metode yang paling sesuai untuk melakukan validasi model simulasi. Perbandingan output simulasi dengan sistem nyata diperlihatkan pada Gambar 3. Jika ukuran kinerja sistem nyata cukup tersedia, uji statistik umum seperti uji t digunakan dimana kita menguji hipotesis kesamaan nilai rata-rata. Kadang-kadang uji F juga dapat digunakan untuk menguji kesamaan ragam sistem nyata

dengan model simulasi. Beberapa metode nonparametrik lainnya juga bisa digunakan, misalnya *ChiSquare* dan *Kolmogorov Smirnov*.



Gambar 3 Perbandingan output simulasi dengan sistem nyata

Perbandingan antara model dan sistem nyata merupakan perbandingan statistik dan perbedaan dalam performa harus diuji untuk signifikansi statistiknya. Perbandingan ini tidak bisa dilakukan dengan sederhana, karena performa yang diukur menggunakan simulasi didasarkan pada periode waktu yang sangat lama, mungkin beberapa tahun. Kinerja yang diukur dalam sistem nyata sebaliknya didasarkan pada periode waktu singkat, mungkin hanya dalam ukuran minggu atau paling lama bulan. Kendala kedua, semua kondisi awal sistem, yang mempunyai pengaruh pada performa sistem secara umum tidak diketahui pada sistem nyata.

Permasalahan lainnya dalam membuat perbandingan statistikal antara sistem nyata dengan model simulasi adalah bahwa performa yang diukur dalam sistem nyata mungkin merefleksikan banyak elemen atau pengaruh dalam sistem yang dikeluarkan dari sistem. Contohnya, ukuran kinerja untuk sistem produksi mungkin memasukkan pengaruh seperti shift kerja panjang, liburan dan kecelakaan industri. Pengaruh ini lebih disukai dikeluarkan dari model simulasi karena pengaruhnya akan konstan untuk sembarang alternatif model simulasi yang diharapkan untuk dievaluasi.

Dalam banyak proyek model yang sedang disimulasikan, sistem nyata bahkan belum ada. Dalam kasus seperti ini, tidak ada ukuran kinerja sistem nyata yang dapat digunakan sebagai perbandingan dengan ukuran kinerja model simulasi. Cara terbaik mungkin mencari sistem yang mirip, tapi perbandingan seperti ini lemah.

- **Metode Delphi**

Metode Delphi dikembangkan sebagai pendekatan ke analisis permasalahan ketika sangat sedikit data tersedia atau sistem nyata sedang dipertimbangkan. Dalam metode Delphi, sekelompok ahli terpilih membentuk panel yang akan menghasilkan jawaban konsensus terhadap pertanyaan yang diajukan ke mereka. Dalam lingkungan simulasi, panel mungkin terdiri dari manager dan pengguna sistem yang sedang dimodekan dan pertanyaan adalah tentang perilaku atau kinerja sistem di bawah kondisi operasi tertentu. Metode Delphi tidak memasukkan diskusi tatap muka, oleh karena itu terhindar dari ketegangan diskusi kelompok seperti dominasi peserta paling vokal. Metode dikembangkan oleh perusahaan RAND dan telah digunakan dalam berbagai bentuk.

Metode Delphi terdiri dari prosedur interaktif berikut :

1. Kuesioner yang memuat pertanyaan respon sistem nyata terhadap input tertentu atau perubahan struktural dikirim ke setiap anggota panel.
2. Didasarkan pada respon akan kuesioner pertama, kuesioner kedua dibentuk yang akan menarik respon lebih spesifik dari panel.
3. Kuesioner baru dikirimkan ke panel bersamaan dengan pemurnian respon panel akan pertanyaan dari tahap sebelumnya.

Tahap 1 sampai 3 diulang 2 kali atau lebih sampai analis mendapatkan prediksi ahli akan respon sistem terhadap input atau perubahan struktural yang sedang dipertimbangkan.

Kritikan pertama terhadap metode Delphi adalah bahwa itu mahal dan memerlukan waktu lama. Hal ini tidak selalu benar. Metode validasi Delphi tidak harus menyebabkan penundaan proyek simulasi karena itu dapat dilakuakn bersamaan dengan pembangunan mode komputer. Metode Delphi mungkin akan sangat mahal, tetapi untuk beberapa proyek simulasi itu bisa relatif lebih murah dibandingkan metode lain.

Kritikan kedua terhadap metode Delphi adalah jika metode itu cukup efektif memprediksi perilaku sistem nyata, mengapa tidak menggunakan metode Delphi ini dalam pemodelan simulasi ?. Dalam beberapa situasi, faktanya metode Delphi mungkin menjadi metode yang efektif biaya; tetapi

bagaimanapun, adalah tidak praktis untuk mempertahankan panel ahli untuk memprediksi respon sistem terhadap perubahan yang sedang dipertimbangkan.

- **Pengujian Turing**

  Metode ini diajukan oleh Alan Turing sebagai uji intelegensia buatan. Seorang ahli atau panel ahli menyediakan ringkasan gambaran atau laporan berdasarkan sistem nyata dan model simulasi. Jika ahli tidak dapat mengidentifikasi laporan berdasarkan output model simulasi, kredibilitas model ditingkatkan. Kesulitan utama validasi model menggunakan uji Turing adalah penyesuaian ukuran kinerja sistem nyata sehingga pengaruh tidak dimaksudkan sebagai bagian dari model simulasi dihilangkan.

- **Perilaku Ekstrim**

  Kadang-kadang sistem nyata dapat diamati di bawah kondisi ekstrim dimana situasi tidak biasa muncul. Kadang-kadang hal ini menjadi solusi ideal untuk mengumpulkan data ukuran kienrja sistem nyata untuk perbandingan output mode simulasi yang dijalankan pada kondisi yang sama. Kadang-kadang juga manager sistem lebih mudah memprediksi bagaimana perilaku sistem nyata pada kondisi ekstrim daripada pada kondisi normal. Dengan membandingkan prediksi perilaku sistem nyata di bawah kondisi ekstrim dengan kinerja model pada kondisi sama, mode dapat divalidasi.

Sargent (1999) menyimpulkan dalam papernya yang berjudul "*Verification and Validation of Simulation Models"* bahwa verifikasi dan validasi model adalah hal yang sangat penting dalam pengembangan sebuah model simulasi. Namun disayangkan, tidak ada pengujian khusus yang dapat dengan mudah diaplikasikan untuk menentukan "kebenaran" dari sebuah model. Lebih jauh, belum ada algoritma yang dapat menentukan teknik atau prosedur apa yang dapat digunakan. Setiap proyek simulasi menghadirkan sebuah tantangan yang unik dan baru.

Referensi

Aninomous. 2011. *Verifikasi dan Validasi Model Simulasi.* http://didi.staff. Guna darma.ac.id/Downloads/files/5040/VERIFIKASI+DAN+VALIDASI+ MODEL+SIMULASI. pdf, diakses pada 1 Oktober 2011.

Cambridge University Press. 2011. *Cambridge Dictionaries Online*. http://dictionary.cambridge.org/ diakses pada 1 Oktober 2011.

Hoover, Perry. 1989. *Simulation A Problem-Solving Approach.* Addison-Wesley., USA.

Law, Kelton. 1991. *Simulation Modeling and Analysis*. McGraw-Hill Inc., *2nd Edition*, New York, USA.

Marimin. 2011. *Materi Kuliah Metode Riset dan Penyajian Ilmiah,* Sekolah Pascasarjana IPB, Bogor.

Sargent RG. 1998. *Verification and Validation of Simulation Models. Proceeding of The 1998 Winter Simulation Conference.* USA. hlm 121-130.

Seminar KB. 2011. *Materi Kuliah Metode Riset dan Penyajian Ilmiah,* Sekolah Pascasarjana IPB, Bogor.

Andi Hasad menempuh pendidikan di program studi Teknik Elektro (S1) UNHAS, Makassar, kemudian melanjutkan di Ilmu Komputer (S2) IPB, Bogor. Penulis pernah menimba ilmu dan pengalaman di berbagai perusahaan / industri di Jakarta dan Bekasi. Saat ini menekuni profesi sebagai dosen tetap di Fakultas Teknik UNISMA Bekasi serta aktif dalam pengembangan ilmu di bidang *robotics, computational intelligence, electronic instrumentation, intelligent control, knowledge management system, network* dan *cryptography*. Info lengkap penulis dapat diakses di http://andihasad.wordpress.com/