BAB II

LANDASAN TEORI

2.1 Sistem Pendukung Keputusan (SPK)

SPK merupakan sistem informasi interaktif yang menyediakan informasi, pemodelan, dan pemanipulasian data. Selain itu sistem pendukung ini banyak digunakan untuk membantu pengambilan keputusan dalam situasi semi terstruktur dan situasi yang tidak terstruktur, dimana tak seorang pun tahu secara pasti bagaimana keputusan seharusnya dibuat. Menurut Turban, et al., 2005, tujuan dari Sistem Pendukung Keputusan adalah sebagai berikut:

1. Membantu manajer dalam pengambilan keputusan atas masalah semi terstruktur.
2. Memberikan dukungan atas pertimbangan manajer dan bukannya di maksudkan untuk menggantikan fungsi manajer
3. Meningkatkan efektivitas keputusan yang di ambil manajer lebih daripada perbaikan efisiensinya
4. Kecepatan komputasi. Komputer memungkinkan para pengambil keputusan untuk melakukan banyak komputasi secara cepat dengan biaya yang rendah
5. Peningkatan produktivitas. Membangun suatu kelompok pengambil keputusan, terutama para pakar, bisa sangat mahal.
6. Dukungan kualitas. Penggunaan komputer mampu meningkatkan kualitas keputusan karena mampu menyimpan banyak data, dapat melakukan simulasi kompleks, mampu memeriksa banyak skenario, cepat, dan ekonomis.
7. Berdaya saing. Adanya persaingan menuntut organisasi untuk mampu secara sering dan cepat mengubah mode operasi, merekayasa ulang proses dan struktur, memberdayakan karyawan, serta berinovasi. Pemanfaatan SPK mampu memecahkan permasalahan tersebut.
8. Mengatasi keterbatasan kognitif dalam pemrosesan dan penyimpanan. Otak manusia memiliki kemampuan yang terbatas untuk memproses dan

menyimpan informasi. Orang-orang kadang sulit mengingat dan menggunakan sebuah informasi dengan cara yang bebas dari kesalahan.

2.2 Karakteristik Sistem Pendukung Keputusan

Beberapa karakteristik SPK antara lain:

1. Sistem pendukung keputusan yang dirancang untuk membantu pengambil keputusan dalam memecahkan masalah yang sifatnya semi terstruktur.
2. Dalam proses pengolahannya, sistem pendukung keputusan mengkombinasikan penggunaan model-model/teknik-teknik analisis dengan teknik pemasukan data konvensional serta fungsi-fungsi pencari/integrasi informasi.
3. Sistem pendukung keputusan dirancang dengan menekankan pada aspek fleksibilitas serta kemempuan adaptasi yang tinggi.
4. Sistem pendukung keputusan, dirancang sedemikian rupa sehingga digunakan/dioperasikan dengan mudah oleh orang-orang yang tidak memiliki dasar kemampuan pengoperasian komputer yang tinggi (Hylenarti, 2018).

2.3 Logika *Fuzzy*

Logika *fuzzy* adalah suatu cara yang tepat untuk memetakan suatu ruang *input* kedalam suatu ruang *output*. Titik awal dari konsep modern mengenai ketidakpastian adalah paper yang dibuat oleh Lofti A Zadeh (1972), dimana Zadeh memperkenalkan teori yang memiliki obyek-obyek dari *himpunan fuzzy* yang memiliki batasan yang tidak presisi dan keanggotaan dalam himpunan *fuzzy*, dan bukan dalam bentuk logika benar (*true*) atau salah (*false*), tapi dinyatakan dalam derajat (*degree*). Konsep seperti ini disebut dengan *Fuzziness* dan teorinya dinamakan *Fuzzy Set Theory*. *Fuzziness* dapat didefinisikan sebagai logika kabur berkenaan dengan semantik dari suatu kejadian, fenomena atau pernyataan itu sendiri. Seringkali ditemui dalam pernyataan yang dibuat oleh seseorang, evaluasi dan suatu pengambilan keputusan.

Fuzzy system (sistem kabur) didasari atas konsep himpunan kabur yang memetakan domain *input* kedalam domain *output*. Perbedaan mendasar

himpunan tegas dengan himpunan kabur adalah nilai keluarannya. Himpunan tegas hanya memiliki dua nilai *output* yaitu nol atau satu, sedangkan himpunan kabur memiliki banyak nilai keluaran yang dikenal dengan nilai derajat keanggotaannya.

Logika *fuzzy* adalah peningkatan dari logika *Boolean* yang berhadapan dengan konsep kebenaran sebagian. Dimana logika klasik (*crisp*) menyatakan bahwa segala hal dapat diekspresikan dalam istilah *binary* (0 atau 1, hitam atau putih, ya atau tidak). Logika *fuzzy* menggantikan kebenaran Boolean dengan tingkat kebenaran. Logika *fuzzy* memungkinkan nilai keanggotaan antara 0 dan 1, tingkat keabuan dan juga hitam dan putih, dan dalam bentuk *linguistic*, konsep tidak pasti seperti "sedikit", "lumayan", dan "sangat". Logika ini diperkenalkan oleh Dr. Lotfi Zadeh dari Universitas California, Barkeley pada tahun 1965.

Logika *fuzzy* telah digunakan pada bidang-bidang seperti taksonomi, topologi, linguistik, teori automata, teori pengendalian, psikologi, *pattern recognition*, pengobatan, hukum, *decision analysis*, system *theory and information retrieval*. Pendekatan *fuzzy* memiliki kelebihan pada hasil yang terkait dengan sifat kognitif manusia, khususnya pada situasi yang melibatkan pembentukan konsep, pengenalan pola, dan pengambilan keputusan dalam lingkungan yang tidak pasti atau tidak jelas.

Ada beberapa alasan mengapa orang menggunakan logika *fuzzy* (Kusumadewi 2003) antara lain:

1. Konsep logika *fuzzy* mudah dimengerti. Konsep matematis yang mendasari penalaran *fuzzy* sangat sederhana dan mudah dimengerti.
2. Logika *fuzzy* sangat fleksibel.
3. Logika *fuzzy* memiliki toleransi terhadap data-data yang tidak tepat.
4. Logika *fuzzy* mampu memodelkan fungsi-fungsi *nonlinear* yang sangat kompleks.

5. Logika *fuzzy* dapat membangun dan mengaplikasikan pengalaman-pengalaman para pakar secara langsung tanpa harus melalui proses pelatihan.
6. Logika *fuzzy* dapat bekerjasama dengan teknik-teknik kendali secara konvensional.
7. Logika *fuzzy* didasarkan pada bahasa alami.

2.3.1 Himpunan Fuzzy

Pada himpunan tegas (*crisp*), nilai keanggotaan suatu item x dalam suatu himpunan A, yang sering ditulis dengan μA[x], memiliki 2 yaitu:

1. Satu (1), yang berarti bahwa suatu item menjadi anggota dalam suatu himpunan, atau
2. Nol (0), yang berarti bahwa suatu item tidak menjadi anggota dalam suatu himpunan.

Terkadang kemiripan antara keanggotaan *fuzzy* dengan probabilitas menimbulkan kerancuan. Keduanya memiliki nilai pada interval [0,1], namun interprestasi nilainya sangat berbeda antara kedua kasus tersebut. Keanggotaan *fuzzy* memberikan suatu ukuran terhadap pendapat atau keputusan, sedangkan probabilitas mengindikasikan proporsi terhadap keseringan suatu hasil bernilai benar dalam jangka panjang. Misalnya, jika nilai keanggotaan bernilai suatu himpunan *fuzzy* USIA adalah 0,9 maka tidak perlu dipermasalahkan berapa seringnya nilai itu diulang secara individual untuk mengharapkan suatu hasil yang hampir pasti muda. Di lain pihak, nilai probabilitas 0,9 usia berarti 10% dari himpunan tersebut diharapkan tidak muda.

Himpunan *fuzzy* memiliki 2 atribut, yaitu:

1. Linguistik, yaitu penamaan suatu grup yang mewakili suatu keadaan atau kondisi tertentu dengan menggunakan bahasa alami, seperti: MUDA, PAROBAYA, TUA

2. Numeris, yaitu suatu nilai (angka) yang menunjukkan ukuran dari suatu variable seperti: 40, 25, 50, dsb.

Ada beberapa hal yang perlu diketahui dalam memahami sistem *fuzzy* (Aplikasi logika *fuzzy* untuk pendukung keputusan, Sri Kusumadewi,Hari Purnomo, Edisi kedua, Graha Ilmu, 2010), yaitu:

a. Variable *fuzzy*

Variable *fuzzy* merupakan variabel yang hendak dibahas dalam suatu sistem *fuzzy*. Contoh: umur, temperature, permintaan, dsb.

b. Himpunan *Fuzzy*

Himpunan *fuzzy* merupakan suatu grup yang mewakili suatu kondisi atau keadaan tertentu dalam suatu variabel *fuzzy*.

c. Semesta Pembicaraan

Semesta pembicaraan adalah keseluruhan nilai yang diperbolehkan untuk dioperasikan dalam suatu variabel *fuzzy*. Semesta pembicaraan merupakan himpunan bilangan *real* yang senantiasa naik (bertambah) secara monoton dari kiri ke kanan. Nilai semesta pembicaraan dapat berupa bilangan positif maupun negatif. Ada kalanya nilai semesta pembicaraan ini tidak dibatasi batas atasnya. Contoh:

a. Semesta pembicaraan untuk variable mahasiswa: [0 50]

b. Semesta pembicaraan untuk variable dosen: [0 50]

d. Domain

Domain himpunan *fuzzy* adalah keseluruhan nilai yang diijinkan dalam semesta pembicaraan dan boleh dioperasikan dalam suatu himpunan *fuzzy*. Seperti halnya semesta pembicaraan, domain merupakan himpunan bilangan *real* yang senantiasa naik (bertambah) secara monoton dari kiri ke kanan. Nilai domain dapat berupa bilangan positif dan bilangan negatif (Purnomo, 2010).

2.3.2 Fungsi Keanggotaan

Fungsi keanggotaan adalah suatu kurva yang menunjukkan pemetaan titik-titik input data ke dalam nilai keanggotaannya (sering juga disebut dengan derajat keanggotaan) yang memiliki interval antara 0 sampai 1. Salah satu

cara yang dapat digunakan untuk mendapatkan nilai keanggotaan adalah dengan melalui pendekatan fungsi. Ada beberapa fungsi yang dapat digunakan

seperti representasi linear naik, linear turun, segitiga, bahu, trapezium, gauss, kurva-S dan lain-lain (Yulmaini, 2015).

2.3.3 Sistem Inferensi Fuzzy Metode Tsukamoto

Metode *Tsukamoto*, implikasi setiap aturan berbentuk implikasi "Sebab-Akibat"/Implikasi "*Input-Output*" dimana antara anteseden dan konsekuen harus ada hubungannya. Setiap aturan direpresentasikan menggunakan himpunan-himpunan *fuzzy,* dengan fungsi keanggotaan yang monoton. Kemudian untuk menentukan hasil **tegas** (*Crisp Solution*) digunakan rumus penegasan (defuzifikasi) yang disebut "Metode rata-rata terpusat" atau "Metode defuzifikasi rata-rata terpusat (*Center Average Deffuzzyfier*) (Kusumadewi, 2004).

Pada metode *Tsukamoto,* setiap konsekuen pada aturan yang berbentuk *IFTHEN* harus direpresentasikan dengan suatu himpunan *fuzzy* dengan fungsi keanggotaan yang monoton. Sebagai hasilnya, *output* hasil inferensi dari tiap-tiap aturan diberikan secara tegas *(crisp)* berdasarkan α-predikat *(fire strength)*. Hasil akhirnya diperoleh dengan menggunakan rata-rata terbobot (Herti, 2018).

Pada *Fuzzy Inference System* (FIS) dikenal beberapa metode yang telah populer, seperti: metode Tsukamoto, metode Mamdani, dan metode Sugeno. Setiap metode memiliki karakteristik yang berbeda. Pada metode Tsukamoto, setiap konsekuen pada aturan yang berbentuk IF-THEN harus direpresentasikan dengan suatu himpunan *fuzzy* dengan fungsi keanggotaan yang monoton. Sebagai hasilnya, *output* hasil inferensi dari tiap – tiap diberikan dengan tegas (*crips*) berdasarkan α-predikat (*fire strength*). Hasil akhirnya diperoleh dengan menggunakan rata-rata terbobot. Secara umum bentuk model *fuzzy Tsukamoto* adalah:

IF(X IS A) and (Y IS B) Then (Z IS C) [3] Dimana A, B, dan C adalah himpunan *fuzzy*. Misalkan diketahui 2 rule beikut.

IF (x is A_1) AND (y is B_1) THEN (z is C_1)

IF (x is A_2) AND (y is B_2) THEN (z is C_2)

Dalam inferensinya, metode Tsukamoto menggunakan tahapan berikut.

1. Fuzzyfikasi
2. Pembentukan basis pengetahuan Fuzzy (Rule dalam bentuk IF…THEN)
3. Mesin Inferensi

   Menggunakan fungsi implikasi MIN untuk mendapatkan nilai αpredikat tiap-tiap rule (α_1, α_2, α_3,.... α_n)
4. Defuzzyfikasi

   Menggunakan metode Rata-Rata (*Average*)

$$z^* = \frac{\sum a_i z_1}{\sum a_i}$$

5. Proses DeFuzzifikasi

Hasil akhir *output* (z) diperoleh dengan menggunakan rata-rata pembobotan:

$$z = \frac{\alpha_1 z_1 + \alpha_2 z_2}{\alpha_1 + \alpha_2}$$

2.4 Perangkat Lunak Yang Digunakan

Komputer membutuhkan perangkat lunak untuk beroperasi dan membutuhkan sistem operasi atau program-program untuk membuat komponen-komponen komputer bekerja secara baik **.** Untuk merancang dan membangun aplikasi ini membutuhkan perangkat lunak penunjang untuk memaksimalkanya, Beberapa perangkat lunak yang digunakan adalah sebagai berikut :

2.4.1 PHP (Hypertext Preproccessor)

Menurut (YM.Khusuma Ardhama, 2012) PHP Hypertext Preprocessor atau sering disebut PHP merupakan bahasa permrograman bebasis server-side yang dapat melakukan parsing script php menjadi menjadi script web sehingga dari sisi client menghasilkan suatu tampilan yang menarik. PHP merupakan pengembangan dari FI atau Form Interface yang dibuat oleh

Rasmus Lerdoff pada tahun 1995. Berbeda dengan HTML, kode PHP tidak diberikan ssecara langsung oleh server ketika ada permintaan atau request dari sisi client namun dengan cara pemrosesan dari sisi server. Kode PHP disisipkan pada kode HTML. Perbedaan dari kode (script) HTML dan PHP yaitu setiap

kode PHP ditulis selalu diberi tag pembuka yaitu <?php dan pada ahir kode PHP diberi tag penutup yaitu ?> .

2.4.2 MySQL

MySQL adalah sebuah program database server yang mampu menerima dan mengirimkan datanya dengan sangat cepat, multi user serta menggunakan perintah standar SQL (Structured Query Language). MySQL juga telah mendukung bahasa pemrograman berfitur API seperti Java sehingga memudahkan para programmer java untuk berkoneksi dengan menggunakan MySQL. (Wahana Komputer, 2012).

2.4.3 XAMPP

Menurut (Herni F & Eri Z, 2012) XAMPP adalah sebuah software *web server* apache yang didalamnya sudah tersedia database *server* MySQL dan dapat mendukung pemrograman PHP. XAMPP merupakan *software* yang mudah digunakan, gratis dan mendukung instalasi di *Linux* dan *Windows*. Keuntungan lainnya adalah cuma menginstal satu kali sudah tersedia *Apache* Web *Server*, MySQL *Database Server*, PHP *Support* (PHP 4 dan PHP 5) dan beberapa *module* lainnya.

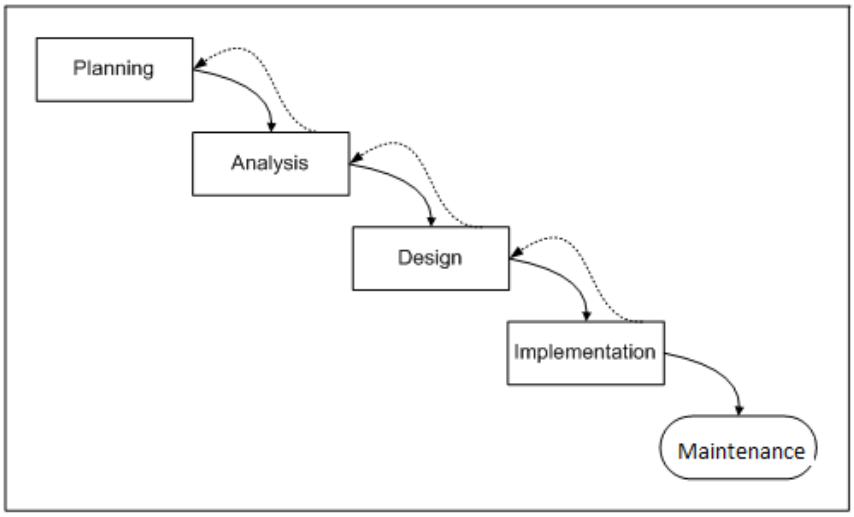
2.5 Website/Web

Menurut Rohi Abdulloh, 2015 dalam jurnal (Sistem Informasi Penjadwalan Dokter Berbassis Web Dengan Menggunakan Framework Codeigniter, 2017) web adalah sekumpulan halaman yang terdiri dari beberapa halaman yang berisi informasi dalam bentuk data digital baik berupa text, gambar, video, audio, dan animasi lainnya yang disediakan melalui jalur koneksi internet. *World wide web* dikembangkan oleh *European Center for Nuclear Research* di Genewa, Swiss sebagai lingkungan yang memungkinkan ilmuwan saling bertukar informasi. *World Wide Web (WWW)* atau W3/W^3 atau cukup disingkat web merupakan kumpulan halaman *hypertext* yang dihubungkan bersama-sama yang menjangkau internet. Ada yang menerjemahkan *WWW* menjadi jaringan jelajah jagad karena menjangkau internet padahal internet sudah

mendunia maka Web pun sudah mendunia, yang meliputi teks, grafik, audio dan video

2.6 Metode Pengembangan Perangkat Lunak

System Development Lyfe Cycle (SDLC) adalah keseluruhan proses dalam membangun sistem melalui beberapa langkah. Model SDLC dan model yang cukup populer dan banyak digunakan adalah waterfall. Menurut (Rosa A.S M.salahuddin, 2015) menjelaskan bahwa Model SDLC air terjun (waterfall) sering juga disebut model eksekuensial linier (sequential linear) atau alur hidup klasik (classiclifecycle). Model waterfall/air terjun menyediakan pendekatan alur hidup perangkat lunak secara sekuensial atau terurut dimulai dari planning, analisis, desain, pengkodean, pengujian dan tahap pendukung (support). Berikut ini adalah gambar model waterfall :



Gambar 2.1. Metode Waterfall

2.7 Unified Modeling Language (UML)

Menurut (Rosa A.S dan Salahudin,2015), *unified modeling language* (UML) adalah salah satu *standar* bahasa yang banyak digunakan di dunia industri untuk mendefinisikan *requirement*, membuat aplikasi dan desain, serta menggambarkan arsitektur dalam pemrograman berorientasi objek.

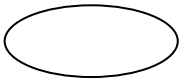
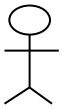
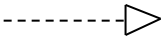
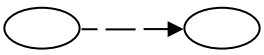
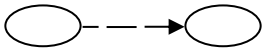
Dengan demikian, penulis dapat mengutarakan bahwa metode *UML* (*Unified Modeling Language*) merupakan sebuah metode atau sebuah bahasa yang

digunakan dalam menterjemahkan, menjelaskan, memodelkan, mendefinisikan suatu sistem dengan bentuk simbol-simbol tertentu yang bertujuan untuk memberikan penjelasan-penjelasan detail dari sebuah sistem.

2.7.1 Use Case Diagram

Menurut (Rosa dan Salahudin,2015), menguraikan *use case* diagram merupakan pemodelan untuk kelakuan sistem informasi yang akan dibuat. *Use case* mendeskripsikan sebuah interaksi antara satu atau lebih aktor dengan sistem informasi yang akan dibuat. *Use case* digunakan untuk mengetahui fungsi apa saja yang ada di dalam sebuah sistem informasi dan siapa saja yang berhak menggunakan fungsi-fungsi itu.

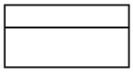
Tabel 2.1. Simbol *Use Case Diagram*

Simbol	**Keterangan**
Use Case 	Menggambarkan bagaimana seseorang akan menggunakan atau memanfaatkan sistem.
Aktor 	Orang, proses atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat diluar sistem informasi yang akan dibuat itu sendiri.
Asosiasi	Komunikasi antara *use case* dan aktor yang berpartisipasi pada *use case* atau *use case* memiliki
Generalisasi 	Sebagai penghubung antara aktor-*use case* atau *use case-use case*.
<<Include>> 	*Include Relationship* (relasi cakupan) : Memungkinkan suatu *use case* untuk menggunakan fungsionalitas yang disediakan oleh *use case* yang
<<Extend>> 	*Extend Relationship* : Memungkinkan relasi *use case* memiliki kemungkinan untuk memperluas fungsionalitas yang

2.7.2 Class Diagram

Class diagram adalah sebuah spesifikasi yang jika diinstansiasi akan menghasilkan sebuah objek dan merupakan inti dari pengembangan dan desainberorientasi objek. *Class* menggambarkan keadaan (atribut/properti) suatu sistem, sekaligus menawarkan layanan untuk memanipulasi keadaan tersebut (metoda/fungsi). *Class diagram* menggambarkan struktur dan deskripsi *class*, *package* dan beserta hubungan satu sama lain seperti *containment*, pewarisan, asosiasi, dan lain-lain.

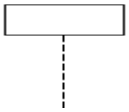
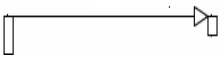
Tabel 2.2. *Simbol Class Diagram*

SIMBOL	NAMA	KETERANGAN
	Generalization	Hubungan dimana objek anak *(descendent)* berbagi perilaku dan struktur data dari objek yang ada di atasnya objek induk (*ancestor*).
	*Nary Association*	Upaya untuk menghindari asosiasi dengan lebih dari 2 objek.
	*Class*	Himpunan dari objek-objek yang berbagi atribut serta operasi yang sama.
	*Collaboration*	Deskripsi dari urutan aksi-aksi yang ditampilkan sistem yang menghasilkan suatu hasil yang terukur bagi suatu actor
	Realization	Operasi yang benar-benar dilakukan oleh suatu objek.
	Dependency	Hubungan dimana perubahan yang terjadi pada suatu elemen mandiri *(independent)* akan mempegaruhi elemen yang bergantung padanya elemen yang tidak mandiri
	Association	Apa yang menghubungkan antara objek satu dengan objek lainnya

2.7.3 Sequence Diagram

Sequence diagram menggambarkan interaksi antar objek di dalam dan di sekitar sistem (termasuk pengguna, *display*, dan sebagainya) berupa pesan yang digambarkan terhadap waktu. *Sequence diagram* terdiri atar dimensi vertikal (waktu) dan dimensi horizontal (objek-objek yang terkait).

Tabel 2.3. *Simbol Class Diagram*

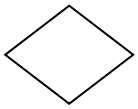
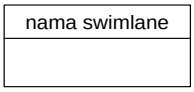
GAMBAR	NAMA	KETERANGAN
	*LifeLine*	Objek *entity*, antarmuka yang saling berinteraksi.
	*Message*	Spesifikasi dari komunikasi antar objek yang memuat informasi-informasi tentang aktifitas yang terjadi
	*Message*	Spesifikasi dari komunikasi antar objek yang memuat informasi-informasi tentang aktifitas.

2.7.4 Activity Diagram

Activity Diagram menggambarkan aliran kerja atau aktivitas dari sebuah sistem atau proses bisnis bukan apa yang dilakukan aktor, jadi aktivitas yang dapat dilakukan oleh sistem.

Tabel 2.4. *Simbol Activity Diagram*

Simbol	Keterangan
Status Awal 	Status awal aktivitas sistem, sebuah diagram aktivitas memiliki sebuah status awal.
Aktivitas 	Aktivitas yang dilakukan sistem, aktivitas biasanya diawali dengan kata kerja.

Percabangan 	Asosiasi percabangan dimana ada pilihan aktivitas lebih dari satu.
Penggabungan	Asosiasi penggabungan dimana lebih dari satu aktivitas digabungkan menjadi satu.
Status Akhir 	Status akhir yang dilakukan sistem, sebuah diagram aktivitas memiliki sebuah status akhir.
Swimlane 	Memisahkan organisasi bisnis yang bertanggung jawab terhadap aktivitas.