**PROYEKSI CURAH HUJAN DAERAH PADANG PARIAMAN  
MENGGUNAKAN DEEP LEARNING DENGAN  
METODE LONG SHORT-TERM MEMORY**

**SKRIPSI**

*Untuk Memenuhi Salah Satu Syarat*

*Memperoleh Gelar Sarjana Komputer*

**Program Studi : Teknik Informatika  
Jenjang Pendidikan : Strata 1 (S1)**



**OLEH :**

**EDO SULAIMAN**  
**NIM. 18101152630092**

**PROGRAM STUDI TEKNIK INFORMATIKA**

**FAKULTAS ILMU KOMPUTER**

**UNIVERSITAS PUTRA INDONESIA “YPTK” PADANG**

**2021**

# BAB II LANDASAN TEORI

## Rekayasa Perangkat Lunak

IEEE Computer Society mendefinisikan Rekayasa Perangkat Lunak (RPL) sebagai penerapan suatu pendekatan yang sistematis, disiplin dan terkuantifikasi atas pengembangan, penggunaan dan pemeliharaan perangkat lunak, serta studi atas pendekatan-pendekatan ini, yaitu penerapan pendekatan engineering atas perangkat lunak (Hasanah & Untari, 2020).

RPL sendiri adalah suatu disiplin ilmu yang membahas semua aspek produksi perangkat lunak, mulai dari tahap awal yaitu analisa kebutuhan pengguna, menentukan spesifikasi dari kebutuhan pengguna, disain, pengkodean, pengujian sampai pemeliharaan sistem setelah digunakan (Hasanah & Untari, 2020).

RPL lebih fokus pada praktik pengembangan perangkat lunak dan mengirimkan perangkat lunak yang bermanfaat kepada pelanggan (customer). Adapun ilmu computer lebih fokus pada teori dan konsep dasar perangkat computer. Rekayasa perangkat lunak lebih fokus pada bagaimana membuat perangkat lunak yang memenuhi kriteria berikut (Hasanah & Untari, 2020) :

1. Dapat terus dipelihara setelah perangkat lunak selesai dibuat seiring berkembangnya teknologi dan lingkungan (maintainability)
2. Dapat diandalkan dengan proses bisnis yang dijalankan perubahan yang terjadi (dependability robust)
3. Efisien dari segi sumber daya dan penggunaan
4. Kemampuan untuk dipakai sesuai dengan kebutuhan (usability)

### Proses Rekayasa Perangkat Lunak

Kerangka kerja proses membangun dasar bagi proses rekayasa perangkat lunak yang lengkap dengan cara mengidentifikasi sejumlah kecil aktivitas kerangka kerja yang cocok bagi semua proyek rekayasa perangkat lunak (Setiyani, 2018). Kerangka kerja proses pada rekayasa perangkat lunak terdiri atas lima aktivitas berikut (Setiyani, 2018) :

#### Komunikasi

Komunikasi, bertujuan untuk memahami tujuan-tujuan stakeholder atas proyek perangkat lunak yang sedang dikembangkan dan mengumpulkan kebutuhan-kebutuhan yang akan membantu mendefinisikan fitur-fitur perangkat lunak berikut dengan fungsi-fungsinya (Setiyani, 2018) :

#### Perencanaan

Kegiatan perencanaan menciptkan suatu peta yang dapat membantu membimbing tim perangkat lunak. Rencana proyek perangkat lunak menggambarkan risiko – risiko yang mungkin muncul, sumber daya yang akan dibutuhkan, produk – produk kerja yang harus dihasilkan dan schedule kerja (Setiyani, 2018) :

#### Pemodelan

Pemodelan, dilakukan bertujuan untuk membuat sketsa sehingga tim perangkat lunak dapat memahami gambaran besar produk yang akan di buat (Setiyani, 2018) :

#### Konstruksi

Konstruksi, sediri adalah kegiatan yang menggabungkan pengkodean dan pengujian (Setiyani, 2018) :

#### Penyerahan

Penyerahan di sini merupakan Penyerahan perangkat lunak kepada user, penyajian perangkat lunak kepada user untuk di evaluasi (Setiyani, 2018) :

### Software Development Life Cycle (SDLC)

System Development Life Cycle (SDLC) adalah metodologi klasik yang digunakan untuk mengembangkan, memelihara dan menggunakan sistem informasi. Siklus hidup sistem itu sendiri merupakan metodologi, tetapi polanya lebih dipengaruhi oleh kebutuhan untuk mengembangkan sistem yang lebih cepat. Pengembangan sistem yang lebih cepat dapat dicapai dengan peningkatan siklus hidup dan penggunaan peralatan pengembangan berbasis computer (Wahyudi, 2018).

#### Tahap-Tahap SDLC

Secara umum tahap-tahap dalam System Development Life Cycle (SDLC)terbagi dalam beberapa tahap (Wahyudi, 2018):

##### Tahap Perencanaan Sistem (system planning)

Tahap Planning Merupakan tahap awal dari pengembangan sistem, tahap ini bertujuan untuk mengidentifikasi dan memprioritaskan sistem informasi apa yang akan dikembangkan, sasaran-sasaran yang ingin dicapai, jangka waktu pelaksanaan serta mempertimbangkan dana yang tersedia dan siapa yang melaksanakan (Wahyudi, 2018).

##### Tahap Analisis Sistem (system analysis)

Tahap Analisis system adalah penelitian atas sistem yang telah ada dengan tujuan untuk merancang system baru atau memperbaharui system yang sudah ada (Wahyudi, 2018).

##### Tahap Perancangan/Desain Sistem (system design)

Tahap Rancangan system adalah penentuan proses dan data yang diperlukan oleh system baru. Jika system ini berbasis komputer, rancangan dapat menyerta kan spesifikasi jenis peralatan yang akan digunakan (Wahyudi, 2018).

##### Tahap Penerapan/Implementasi Sistem (system implementation)

Tahap Penerapan merupakan kegiatan memperoleh dan mengintegrasikan sumber daya fisik dan konseptual yang menghasilkan suatu sistem yang bekerja. Pada tahapan ini dilakukan beberapa hal yaitu: Coding, Testing, Instalasi. Dan Output dari tahapan ini adalah : source code, prosedur, pelatihan (Wahyudi, 2018).

##### Tahap Pemeliharaan/Perawatan Sistem

Tahap pemeliharaan/perawatan sistem merupakan tahap yang dilakukan setelah tahap implementasi yang meliputi penggunaan sistem, audit sistem, penjagaan sistem, perbaikan sistem dan peningkatan sistem (Wahyudi, 2018).

#### Model Waterfall

Model SDLC air terjun (waterfall) sering juga disebut model sekuensial linier (sequential linier) atau alur hidup klasik (classic life cycle). Model air terjun menyediakan pendekatan alur hidup perangkat lunak secara sekuential atau terurut dimulai dari analisis kebutuhan perangkat lunak, desain, pembuatan kode program, pengujian, dan pemeliharaan (maintenance) (Tabrani & Pudjiarti, 2021). Berikut tahap tahapanya sebagai beikut (Tabrani & Pudjiarti, 2021):

##### Analisis Kebutuhan Perangkat Lunak

Proses pengumpulan kebutuhan dilakukan secara intensif untuk menspesifikasikan kebutuhan perangkat lunak agar dapat dipahami perangkat lunak seperti apa yang dibutuhkan oleh user. Spesifikasi kebutuhan perangkat lunak pada tahap ini perlu untuk didokumentasikan (Tabrani & Pudjiarti, 2021).

##### Desain

Desain perangkat lunak adalah proses multi langkah yang fokus pada desain pembuatan program perangkat lunak termasuk struktur data, arsitektur perangkat lunak, representasi antarmuka, dan prosedur pengkodean. Tahap ini mentrannslasi kebutuhan perangkat lunak dari tahap analisis kebutuhan ke representasi desain agar dapat diimplementasikan menjadi program pada tahap selanjutnya. Desain perangkat lunak yang dihasilkan pada tahap ini juga perlu didokumentasikan (Tabrani & Pudjiarti, 2021).

##### Pembuatan Kode Program

Desain harus ditranslasikan kedalam program perangkat lunak. Hasil dari tahap ini adalah program komputer sesuai dengan desain yang telah dibuat pada tahap desain (Tabrani & Pudjiarti, 2021).

##### Pengujian

Pengujian fokus pada perangkat lunak secara dari segi lojik dan fungsional dan memastikan bahwa semua bagian sudah diuji. Hal ini dilakukan untuk meminimalisir kesalahan (error) dan memastika keluaran yang dihasilkan sesuai dengan yang diinginkan (Tabrani & Pudjiarti, 2021).

##### Pendukung atau Pemeliharaan (maintenance)

Tidak menutup kemungkinan sebuah perangkat lunak mengalami perubahan ketika sudah dikirimkan ke user. Perubahan bisa terjadi karena adanya kesalahan yang muncul dan tidak terdeteksi saat pengujian atau perangkat lunak harus beradaptasi dengan lingkungan baru. Tahap pendukung atau pemeliharaan dapat mengulangi proses pengembangan mulai dari analisis spesifikasi untuk perubahan perangkat lunak yang sudah ada, tapi tidak untuk membuat perangkat lunak baru (Tabrani & Pudjiarti, 2021).

## Unified Modelling Language *(UML)*

Pemodelan dalam suatu rekayasa perangkat lunak merupakan suatu hal yang dilakukan di tahapan awal. Pemodelan delam perangkat lunak merupakan suatu yang harus dikerjakan di bagian awal dari rekayasa, dan pemodelan ini akan mempengaruhi perkerjaan-pekerjaan dalam rekayasa perangkat lunak tersebut. Salah satu perangkat pemodelan adalah Unified Modelling Language (UML). UML merupakan salah satu standart bahasa yang banyak digunakan di dunia industry untuk mendefinisikan requirement, membuat analisis & desain, sert amenggambarkan arsitektur dalam pemrograman berorientasi objek. UML muncul karena adanya kebutuhan pemodelan visual untuk menspesifikasikan, menggambarkan, membangun, dan mendokumentasikan sistem perangkat lunak (Hasanah & Untari, 2020).



(Sumber : Hasanah & Untari, 2020)*.*

Gambar 2. Bagan UML

Berdasarkan **Gambar 2** berikut penjelasan singkat dari pembagian kategori tersebut (Hasanah & Untari, 2020) :

1. Behavior diagram, merupakan kumpulan diagram yang digunakan untuk menggambarkan kelakuan sistem atau rangkaian perubahan yang terjadi pada suatu sistem.
2. Interaction diagram, merupakan kumpulan diagram yang digunakan untuk menggambarkan interaksi sistem dengan sistem lain maupun interaksi antar subsistem pada suatu sistem.
3. Structure diagram, merupakan kumpulan diagram yang digunakan untuk menggambarkan strutur statis dari sistem yang dimodelkan.

### Behavior Diagrams

#### Use Case Diagram

Use case diagram adalah teknik untuk merekam persyaratan fungsional sebuah system, menggambarkan fungsionalitas yang diharapkan dari sebuah sistem. Use case diagram menekankan kepada “apa” yang diperbuat oleh sistem, dan bukan “bagaimana”. Sebuah use case merepresentasikan sebuah interaksi antara actor dengan sistem. Seorang atau sebuah aktor adalah sebuah entitas dapat berupa manusia atau mesin yang berinteraksi dengan system untuk melakukan pekerjaan-pekerjaan tertentu. (Hasanah & Untari, 2020).

Table 1. Notasi Use Case Diagram

|  |  |  |
| --- | --- | --- |
| No | Simbol | Keterangan |
| 1 | *Use case* | Fungsionalitas yang disediakan sistem sebagai unit-unit yang saling bertukar pesan antar unit atau aktor. |
| 2 | *Actor*  *Description: Description: actor* | Orang, proses atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat di luar sistem itu. |
| 3 | *Association* | use case yang memiliki interaksi dengan aktor. |
| 4 | *Extend* | Relasi use case tambahan ke sebuah use case dimana use case yang ditambahkan dapat berdiri sendiri. |
| 5 | *Generalization* | Relasi use case tambahan ke sebuah use case dimana use case yang ditambahkan memerlukan ini untuk menjalankan fungsinya. |

(Sumber : Hasanah & Untari, 2020)

#### Activity Diagram

Activity diagram menggambarkan berbagai alir aktivitas dalam sistem yang sedang dirancang, bagaimana masing-masing alir berawal, decision yang mungkin terjadi, dan bagaimana mereka berakhir. Activity diagram digunakan untuk menggambarkan langkahlangkah atau aktivitas pada suatu sistem (Hasanah & Untari, 2020).

Table 2. Notasi Activity Diagram

|  |  |  |
| --- | --- | --- |
| No | Simbol | Keterangan |
| 1 | *Initial State* | Sebuah diagram aktivitas memiliki sebuah status awal |
| 2 | *Activity* | Aktivitas yang dilakukan sistem, aktivitas biasanya diawali dengan kata kerja |
| 3 | *Decision* | Asosiasi percabangan dimana jika ada pilihan aktivitas lebih dari satu |
| 4 | *Join* | Asosiasi penggabungan dimana lebih dari satu aktivitas digabungkan menjadi satu |
| 5 | *Final State* | Status akhir yang dilakukan sistem, sebuah diagram aktivitas memiliki sebuah status akhir. |
| 6 | *Swimlane*    or | Memisahkan organisasi bisnis yang bertanggung jawab terhadap aktivitas yang terjadi. |

(Sumber : Hasanah & Untari, 2020)

#### Statechart Diagram

Statechart diagram menelusuri individu-individu objek melalui keseluruhan daur hidupnya, menypesifikasikan semua urutan yang mungkin dari pesan-pesan yang akan diterima objek tersebut, bersama-sama dengan tanggapan atas pesan-pesan tersebut (Ilham & Fajri, 2020).

Table 3. Notasi Statechart Diagram

|  |  |  |
| --- | --- | --- |
| No | Simbol | Keterangan |
| 1 | *Initial State* | Kelas pada struktur sistem |
| 2 | *Final State* | Sama dengan konsep interface dalam beberapa pemrograman berorientasi objek. |
| 3 | *Event* | Relasi antarkelas dengan makna umum.  Asosiasi biasanya juga disertai dengan multiplicity |
| 4 | *State* | Relasi antarkelas dengan makna kelas yang satu digunakan oleh kelas yang lain. |

(Sumber : Ilham & Fajri, 2020).

### Interaction Diagrams

#### Sequence Diagram

Sequence diagrammenggambarkan interaksi antar objek di dalam dan di sekitar sistem (termasuk pengguna, display, dan sebagainya) berupa messageyang digambarkan terhadap waktu. Sequence diagramterdiri atas dimensi vertikal (waktu) dan dimensi horizontal (objek-objek yang terkait). Sequence diagrambiasa digunakan untuk menggambarkan skenario atau rangkaian langkah-langkah yang dilakukan sebagai respons dari sebuah eventuntuk menghasilkan outputtertentu. Diawali dari apa yang men-triggeraktivitas tersebut, proses dan perubahan apa saja yang terjadi secara internal dan outputapa yang dihasilkan (Rinaldi, 2019).

Table 4. Notasi State Machine Diagram

|  |  |  |
| --- | --- | --- |
| No | Simbol | Keterangan |
| 1 | *Actor*  *Description: Description: actor*  Or  Nama aktor | Orang, proses, atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat di luar sistem informasi biasanya dinyatakan menggunakan kata benda di awal frasa nama actor |
| 2 | *Lifeline* | Menyatakan kehidupan suatu objek |
| 3 | *Object*  objek : kelas | Menyatakan objek yang berinteraksi pesan |
| 4 | *Active Time* | Menyatakan objek dalam keadaan aktif berinteraksi, semua yang terhubung dengan waktu aktif ini adalah sebuah tahapan yang dilakukan di dalamnya, |
| 5 | *Message type C****reate***  <<*create*>> | Menyatakan suatu objek membuat objek yang lain, arah panah mengarah pada objek yang dibuat |
| 6 | *Message type* ***Call***  1:nama\_metode() | Menyatakan suatu objek memanggil operasi/metode yang ada pada objek lain atau dirinya sendiri |
| 7 | *Message type* ***Send***  1: masukan | Menyatakan bahwa suatu objek mengirimkan data/masukan/informasi ke objek lainnya, arah panah mengarah pada objek yang dikirim |
| 8 | *Message type* ***Return***  1: keluaran  ---------------------- | Menyatakan bahwa suatu objek yang telah menjalankan suatu operasi atau metode menghasilkan suatu kembalian ke objek tertentu, arah panah mengarah pada objek yang menerima kembalian |
| 9 | *Message type* ***Destroy***  <<*destroy*>> | Menyatakan suatu objek mengakhiri hidup objek yang lain, arah panah mengarah pada objek yang diakhiri, jika ada create maka ada destroy. |

(Sumber : Rinaldi, 2019)

#### Collaboration Diagram

Collaboration diagram adalah cara alternative untuk mengetahui tahap-tahap terjadinya suatu aktivitas. Perbedaan antara collaboration dan sequence diagram adalah collaboration diagram memperlihatkan bagaimana hubungan antara beberapa objek berdasarkan urutan dari pesan, sedangkan sequence diagram memperlihatkan bagaimana urutan kejadian berdasarkan waktu (Ilham & Fajri, 2020).

Table 5. Notasi Collaboration Diagram

|  |  |  |
| --- | --- | --- |
| No | Simbol | Keterangan |
| 1 | *Object*  objek : kelas | Objek yang melakukan interaksi pesan. |
| 2 | *Link* | Relasi antara objek yang menghubungkan objek satu dengan lainya atau dengan diri sendiri. |
| 4 | *State* | Relasi antar kelas dengan makna kelas yang satu digunakan oleh kelas yang lain. |

(Sumber : Ilham & Fajri, 2020)

### Structure Diagrams

#### Class Diagram

Class diagram adalah sebuah spesifikasi yang jika diinstansiasi akan menghasilkan sebuah objek dan merupakan inti dari pengembangan dan desain berorientasi objek. Class diagram menggambarkan keadaan (atribut/properti) suatu sistem, sekaligus menawarkan layanan untuk memanipulasi keadaan tersebut (metode/fungsi). Class diagram menggambarkan struktur dan deskripsi class, package dan objek beserta hubungan satu sama lain seperti containment, pewarisan, asosiasi, dan lain-lain (Rinaldi, 2019).

Tabel 6. Notasi Class Diagram

|  |  |  |
| --- | --- | --- |
| No | Simbol | Keterangan |
| 1 | *Class*  **nama\_kelas**  + atribut  + operasi() | Kelas pada struktur sistem |
| 2 | *Interface* | Sama dengan konsep interface dalam beberapa pemrograman berorientasi objek. |
| 3 | *Association* | Relasi antar kelas dengan makna umum.  Asosiasi biasanya juga disertai dengan multiplicity |
| 4 | *Directed Association* | Relasi antar kelas dengan makna kelas yang satu digunakan oleh kelas yang lain. |
| 5 | *Dependency* | Relasi antar kelas dengan ketergantungan antar kelas |
| 6 | *Aggregation* | Relasi antar kelas dengan makna semua-bagian (whole-part) |

(Sumber : Rinaldi, 2019)

#### Deployment Diagram

Menggambarkan secara lengkap bagaimana komponen di deployment dalam infrastruktur sistem, dimana komponen akan terletak, bagaimana kemampuan jaringan pada kondisi tertentu, spesifikasi server, dan hal-hal lain yang bersifat fisikal (Wijaya, Masriadi, & Ikhlas, 2020).

Tabel 7. Notasi Deployment Diagram

|  |  |  |
| --- | --- | --- |
| No | Simbol | Keterangan |
| 1 | *Package*  [Description: http://4.bp.blogspot.com/-sc_hs41py7c/VR5gbgH2yHI/AAAAAAAAAXU/xAgOjf7Nc-g/s1600/Snap%2B2015-04-03%2Bat%2B16.41.02.png](http://4.bp.blogspot.com/-sc_hs41py7c/VR5gbgH2yHI/AAAAAAAAAXU/xAgOjf7Nc-g/s1600/Snap+2015-04-03+at+16.41.02.png) | Package merupakan sebuah bungkusan dari satu atau lebih komponen. |
| 2 | *Node*  Nama\_node | Biasanya mengacu pada perangkat keras(hardware), perangkat lunak yang tidak dibuat sendiri (software), jika didalam node disertakan komponen untuk mengkonsistensikan rancangan maka komponen yang diikutsertakan harus sesuai dengan komponen yang telah didefinisikan sebelumnya pada diagram komponen. |
| 4 | *Dependency* | Ketergantungan antar komponen, arah panah mengarah pada komponen yang dipakai. |
| 5 | *Link* | Relasi antar komponen. |

(Sumber : Wijaya, Masriadi, & Ikhlas, 2020).

## Artificial Intelligence

*Andreas Kaplan* dan *Michael Haenlein* mendefinisikan kecerdasan buatan / *Artificial Intelligence (AI)* sebagai kemampuan system untuk menafsirkan data eksternal dengan benar, untuk belajar dari data tersebut, dan menggunakan pembelajaran tersebut guna mencapai tujuan dan tugas tertentu melalui adaptasi yang fleksibel. Sistem seperti ini umumnya dianggap sebagai computer (Siahaan, et al., 2020).

Lebih lanjut *Budiharto* menyatakan bahwa *Intelligence* merupakan istilah yang kompleks yang dapat didefinisikan dengan ungkapan yang berbeda seperti logika, pemahaman, self-awareness, pembelajaran, perencanaan, dan problem solving. Sedangkan “Artificial” adalah sesuatu yang tidak nyata, seperti tipuan karena merupakan hasil simulasi (Sihombing & Syaputra, 2020).

Savitri menguraikan bahwa Kecerdasan buatan / Artificial intelligence (AI) merupakan bidang ilmu komputer yang menekankan pada penciptaan mesin cerdas yang bekerja dan bereaksi seperti manusia yang perkembangannya terjadi sangat pesat di era revolusi industri keempat (Sihombing & Syaputra, 2020).

Kecerdasan diciptakan dan dimasukkan kedalam suatu mesin / komputer agar dapat melakukan pekerjaan seperti yang dapat dilakukan oleh manusia. Beberapa macam bidang yang menggunakan kecerdasan buatan antara lain sistem pakar, permainan komputer (games), logika fuzzy, jaringan saraf tiruan dan robotika (Siahaan, et al., 2020).

Tujuan utama dari pembuatan AI adalah untuk membuat mesin memiliki fungsi yang memiliki kriteria-kriteria kecerdasan didalamnya sehingga mesin tersebut mampu melakukan pekerjaan manusia yang lebih kompleks, tergantung dari tingkat kecerdasan AI yang digunakan (Gunova, 2021).

Dalam penerapannya, terdapat 6 kemampuan utama yang dapat diklasifikasikan sebagai AI (Gunova, 2021), Kemampuan tersebut antara lain :

* Representasi Informasi/Pengetahuan (Knowledge Representation)
* Perencanaan (Planning)
* Persepsi (Perception)
* Machine Learning / Deep Learning
* Pemahaman Bahasa (Natural Language Processing)
* Robotics

Meskipun 6 kemampuan diatas memiliki fungsi yang berbeda-beda, kemampuan kemampuan tersebut saling berhubungan satu sama lain saat pengaplikasiaannya. Bahkan hubungan antar kemampuan utama diatas akan membentuk kemampuan baru (Gunova, 2021).

## Machine Learning

Machine Learning merupakan Disiplin ilmu yang menggunakan berbagai pendekatan untuk mengajarkan komputer untuk menyelesaikan tugas-tugas di mana tidak ada algoritma yang sepenuhnya memuaskan tersedia. Dalam kasus di mana terdapat sejumlah besar jawaban potensial, satu pendekatan adalah memberi label beberapa jawaban yang benar sebagai valid. Ini kemudian dapat digunakan sebagai data pelatihan bagi komputer untuk meningkatkan algoritme yang digunakannya untuk menentukan jawaban yang benar. Misalnya, untuk melatih sistem untuk tugas pengenalan karakter digital, dataset MNIST dari angka tulisan tangan sering digunakan (Alpaydin, 2020).

Program *Machine Learning* dapat melakukan tugas tanpa diprogram secara eksplisit untuk melakukannya. Ini melibatkan komputer belajar dari data yang disediakan sehingga mereka melakukan tugas-tugas tertentu. Untuk tugas-tugas sederhana yang ditugaskan ke komputer, dimungkinkan untuk memprogram algoritme yang memberi tahu mesin bagaimana menjalankan semua langkah yang diperlukan untuk memecahkan masalah yang dihadapi; di bagian komputer, tidak diperlukan pembelajaran. Untuk tugas yang lebih maju, mungkin sulit bagi manusia untuk membuat algoritme yang diperlukan secara manual. Dalam praktiknya, ternyata lebih efektif untuk membantu mesin mengembangkan algoritmenya sendiri, daripada meminta pemrogram manusia menentukan setiap langkah yang diperlukan (Alpaydin, 2020).

Algoritma *Machine Learning* digunakan dalam berbagai macam aplikasi, seperti dalam kedokteran, penyaringan email, pengenalan suara, dan visi komputer, di mana sulit atau tidak mungkin untuk mengembangkan algoritma konvensional untuk melakukan tugas-tugas yang diperlukan (Hu, Niu, Carrasco, Lennox, & Arvin, 2020).

## Deep Learning

Sejak tahun 2006, *Deep Structured Learning* atau yang lebih dikenal dengan *Deep learning* atau *Hierarchical Learning* telah muncul sebagai area baru dalam penelitian *Machine Learning* yang berdasarkan pada suatu set algoritma yang mencoba untuk memodelkan abstraksi tingkat tinggi pada data dengan menggunakan graf yang mendalam dengan beberapa layer pengolahan, yang terdiri dari beberapa transformasi linier dan non-linier (François, 2018).

*Deep Learning* sendiri adalah cabang ilmu *machine learning* berbasis *Neural Network (NN)* atau bisa dikatakan sebagai perkembangan dari *Neural Network* (Ilahiyah & Nilogiri, 2018). Metode pendekatan *deep learning* mengklasifikasi data dalam dua sesi yaitu sesi training dan testing. Pada sesi training mempelajari ekstrasi fitur dari setiap data supaya bisa membedakan suaru label dengan label yang lain. Pada sesi testing data-data yang diuji dapat dianalisa dari hasil sesi training (Azizah, Umayah, & Fajar, 2018).



(Sumber : Savalia & Emamian, 2018)

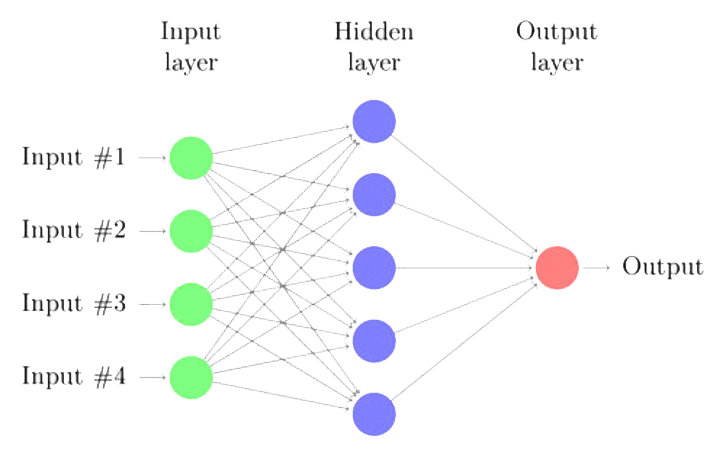
Gambar 3. Perbedaan Simple Neural Network dan Deep Learning

Dalam *deep learning*, sebuah komputer belajar mengklasifikasi secara langsung dari gambar atau suara. Metode *deep learning* menggunakan CPU dan RAM dalam proses komputasi, dan juga memanfaatkan GPU sehingga proses komputasi data yang besar dapat berlangsung lebih cepat (Ilahiyah & Nilogiri, 2018).

## Jaringan Syaraf Tiruan

*Jaringan saraf tiruan (JST) / Artificial Neural Network (ANN) / Neural Nenwork (NN)* adalah jaringan komputasi terinspirasi dari cara kerja otak manusia karena terbukti Otak manusia melakukan sesuatu hal yang sama persis seperti, hirarki pertama pada neuron menerima formasi pada visual cortex yang sensitif terhadap gambaran tepi dan gumpalan khusus (François, 2018).

Menurut Qomariyah, Hamzah, & Mustika *Artificial Neural Network* merupakan metode pemrosesan informasi yang mempunyai karakteristik menyerupai jaringan syaraf biologi. Bagian terpenting dari Artificial Neural Network adalah kemampuannya dalam melakukan pembelajaran (Qomariyah, Hamzah, & Mustika, 2018).



(Sumber : Shekar, S´a, Ferreira, & Soares, 2018)

Gambar 4. Skema Jaringan Saraf Tiruan

Seperti yang terlihat pada **Gambar 4** Lapisan-lapisan penyusun JST tersebut dapat dibagi menjadi 3, yaitu (Satria, 2018) :

1. Lapisan Input, unit-unit di dalam lapisan input disebut unit-unit input. Unit-unit input tersebut menerima pola inputan data dari luar yang menggambarkan suatu permasalahan (Satria, 2018)
2. Lapisan tersembunyi, unit -unit di dalam lapisan tersembunyi disebut unit- unit tersembunyi (Satria, 2018)
3. Lapisan output, unit-unit didalam lapisan outputdisebut unit-unit output (Satria, 2018)

## Long Short-Term Memory

*Long Short-Term Memory (LSTM)* merupakan pengembangan dari *Recurrent Neural Network (RNN)* dengan mengatasi salah satu kekurangan *RNN* yaitu kemampuan pengelolaan informasi dalam periode yang lama yang mana dilakukan modifikasi pada Recurrent Neural Network (RNN) dengan memberi memory cell untuk dapat menyimpan informasi dalam waktu yang lama. Diusulkan oleh Sepp Hochreiter dan Jurgen Schmidhuber pada tahun 1997, *LSTM* banyak dipilih untuk prediksi berbasis waktu atau *time series* karena dikenal lebih unggul dan handal dalam melakukan prediksi dalam waktu lama dibanding algoritma lain (Zahara, Sugianto, & Ilmiddafiq, 2019; Manaswi, 2018).

A screenshot of a video game

Description automatically generated with medium confidence  
Diagram

Description automatically generated

(Sumber : Manu, 2021)

Gambar 5. Perulangan modul dalam LSTM berisi empat layer yang saling berinteraksi.

Kunci LSTM adalah *cell state*, garis horizontal yang melewati bagian atas diagram keadaan sel seperti ban berjalan. Ini berjalan lurus ke bawah seluruh rantai, memiliki beberapa linier kecil interaksi (Ghosh, Bose, Maji, Debnath, & Sen, 2019). Untuk setiap sel memori memiliki tiga layer sigmoid dan satu layer tanh (Qiu, Wang, & Zhou, 2019).

A picture containing text, clock

Description automatically generated

(Sumber : Manu, 2021)

Gambar 6. Alur Informasi Cell State pada LSTM

Keterangan (Pardede & Ibrahim, 2020) :

: *Cell state*

: Nilai *Cell state* sebelum order ke *t*

Pada **Gambar 6**, garis horizontal yang melalui bagian atas diagram dikenal sebagai *cell state* (, ). Ini bertindak seperti ban berjalan yang berjalan di seluruh jaringan. Ini membawa informasi dari sel sebelumnya ke saat ini dan seterusnya (Hiransha, Gopalakrishnan, Menon, & Soman, 2018). Kemampuan untuk menambah atau menghapus informasi ke cell state dikendalikan oleh struktur yang disebut *gate*. *Gate* digunakan untuk secara opsional membiarkan informasi lewat. Informasi yang di saring melalui struktur *gate* yang akan mempertahankan dan memperbarui *cell state* memori, LSTM memiliki tiga di antaranya gerbang, untuk melindungi dan mengontrol cell state, Struktur gerbangnya mencakup *forget gate*, *input gate.* dan *output gate*. *Gate* terdiri dari layer jaring saraf sigmoid dan operasi perkalian pointwise (Ghosh, Bose, Maji, Debnath, & Sen, 2019; Qiu, Wang, & Zhou, 2019).

Graphical user interface, application

Description automatically generated

(Sumber : Manu, 2021)

Gambar 7. Layer sigmoid mengeluarkan angka antara nol dan satu.

Pada **Gambar 7** layer sigmoid mengeluarkan angka antara 0 dan 1, menggambarkan berapa banyak dari setiap komponen yang harus dilewati. Nilai 0 berarti “jangan biarkan apa pun lewat”, sedangkan nilai 1 berarti “biarkan semuanya lewat!” (Ghosh, Bose, Maji, Debnath, & Sen, 2019)*.* Berikut ialah keterangan setiap gate yang ada pada **Gambar 5** :

### Forget Gate

Pada *forget gate* informasi pada setiap data *input* yang akan diolah dan dipilih data mana saja yang akan disimpan atau dibuang pada memory cells. Fungsi aktivasi yang digunakan pada *forget gate* ini adalah fungsi aktivasi sigmoid. Dimana hasil keluarannya antara 0 dan 1. Jika keluarannya adalah 1 maka semua data akan disimpan dan sebaliknya jika keluarannya 0 maka semua data akan dibuang (Aldi, Jondri, & Aditsania, 2018). Dengan rumus seperti pada **Gambar 8** :

Diagram

Description automatically generated

(Sumber : Manu, 2021)

Gambar 8. Alur Informasi Forget Gate pada LSTM

Keterangan (Pardede & Ibrahim, 2020) :

: *Forget gate*

: Fungsi Aktivasi *Sigmoid*

: Nilai *Weight* untuk *Forget gate*

: Nilai *output* sebelum order ke *t*

: Nilai *input* pada order ke *t*

: Nilai bias pada *Forget gate*

Pada **Gambar 8** LSTM memutuskan informasi apa yang akan dibuang dari *cell state*. Keputusan ini dibuat oleh layer sigmoid yang disebut "layer forget gate." (). Terlihat pada dan dan nilai *output* antara angka 0 dan 1 untuk setiap angka dalam *cell state* pada **Gambar 6**. *Output* dari 1 mewakili 'sepenuhnya simpan ini' sementara 0 mewakili 'singkirkan ini sepenuhnya' (Boruah & Barman, 2018).

### Input Gate

Pada *input gate* terdapat dua gates yang akan dilaksanakan, pertama akan diputuskan nilai mana yang akan diperbarui menggunakan fungsi aktivasi sigmoid. Selanjutnya fungsi aktivasi tanh akan membuat vector nilai baru yang akan disimpan pada memory cell (Aldi, Jondri, & Aditsania, 2018). Dengan rumus seperti pada **Gambar 9** :

Diagram

Description automatically generated

(Sumber : Manu, 2021)

Gambar 9. Alur Informasi yang melewati Input Gate pada LSTM

Keterangan (Pardede & Ibrahim, 2020) :

: *Input gate*

: Nilai baru yang dapat ditambahkan ke *cell state*

: Fungsi Aktivasi *Sigmoid*

: Fungsi Aktivasi *Tanh*

: Nilai *Weight* untuk *Input gate*

: Nilai *Weight* untuk *Cell state*

: Nilai *output* sebelum order ke *t*

: Nilai *input* pada order ke *t*

: Nilai bias pada *Input gate*

: Nilai bias pada *cell state*

Pada langkah berikutnya di **Gambar 9** LSTM memutuskan informasi apa yang akan disimpan dari *cell state*. Pertama layer *sigmoid* yang disebut "layer *input gate*" () memutuskan nilai mana yang akan diperbarui. Setelah itu, layer membuat vektor nilai kandidat baru, , yang dapat ditambahkan ke state (Ghosh, Bose, Maji, Debnath, & Sen, 2019).

### Cell State / Memory State

Pada *cell state* gates akan mengganti nilai pada memory cell sebelumnya dengan nilai memory cell yang baru. Dimana nilai ini didapatkan dari menggabungkan nilai yang terdapat pada forget gate dan input gate (Aldi, Jondri, & Aditsania, 2018). Dengan rumus seperti pada **Gambar 10** :

Diagram

Description automatically generated with medium confidence

(Sumber : Manu, 2021)

Gambar 10. Alur Memperbaharui Cell State pada LSTM

Keterangan (Pardede & Ibrahim, 2020) :

: *Cell state*

: *Forget gate*

: Nilai *Cell state* sebelum order ke *t*

: *Input gate*

: Kandidat konteks baru yang dapat ditambahkan ke *cell state*

Pada langkah selanjutnya di **Gambar 10**, kedua layer di gabungkan digabungkan untuk membuat pembaruan ke *cell state*. Pada Langkah inilah nilai *cell state* lama (), akan di perbaharui ke nilai dari *cell state* baru () dimana LSTM akan mengalikan *cell state* lama dengan () kemudian ditambahkan dengan () . Ini adalah nilai kandidat baru, yang diskalakan berdasarkan seberapa banyak kami memutuskan untuk memperbarui setiap nilai negara bagian. (Ghosh, Bose, Maji, Debnath, & Sen, 2019).

### Output Gate

Pada *output gate* terdapat dua gate yang akan dilaksanakan, pertama akan diputuskan nilai pada bagian memory cell mana yang akan dikeluarkan dengan menggunakan fungsi aktivasi sigmoid. Selanjutnya akan ditempatkan nilai pada memory cell dengan menggunakan fungsi aktivasi tanh. Terakhir kedua gate tersebut di dikalikan sehingga menghasilkan nilai yang akan dikeluarkan (Aldi, Jondri, & Aditsania, 2018). Dengan rumus seperti pada **Gambar 11** :

A picture containing text, clock

Description automatically generated

(Sumber : Manu, 2021)

Gambar 11. Alur Informasi melewati Output Gate pada LSTM

Keterangan (Pardede & Ibrahim, 2020) :

: *Output gate*

: *Cell state*

: Fungsi Aktivasi *Sigmoid*

: Fungsi Aktivasi *Tanh*

: Nilai *Weight* untuk *Output gate*

: Nilai *output* sebelum order ke *t*

: Nilai *input* pada order ke *t*

: Nilai bias pada *Output gate*

: Nilai *output* pada order ke *t*

Terakhir pada **Gambar 11** adalah tahap dimana perlu memutuskan apa yang akan hasilkan. *Output* akan didasarkan pada *cell state*, tetapi akan menjadi versi yang difilter. Pertama, kita menjalankan layer sigmoid yang memutuskan bagian mana dari *cell state* yang akan kita hasilkan. Kemudian, *cell state* di tempatkan melalui *tanh* (untuk mendorong nilai menjadi antara -1 dan 1) dan mengalikannya dengan *output* *gate* layer *sigmoid*, sehingga hanya akan menampilkan bagian yang putuskan (Ghosh, Bose, Maji, Debnath, & Sen, 2019).

## Prediksi / Forecasting

Menurut Sucipto & Syaharuddin Prediksi / Peramalan (forecasting) adalah kegiatan mengestimasi apa yang akan terjadi pada masa yang akan datang. Peramalan diperlukan karena adanya kesenjangan waktu (timelag) antara kesadaran dibutuhkannya suatu kebijakan baru dengan waktu pelaksanaan kebijakan tersebut (Sucipto & Syaharuddin, 2018).

Menurut Putro, Furqon, & Wijoyo prediksi merupakan suatu proses untuk meramalkan atau memperkirakan suatu variable di masa yang akan datang. Dalam kasus prediksi biasanya data yang sering digunakan adalah data kuantitatif. Prediksi tidak harus menghasilkan suatu jawaban yang pasti kejadian, melainkan berusaha untuk mencari jawaban yang sedekat mungkin dengan kejadian yang akan terjadi (Putro, Furqon, & Wijoyo, 2018).

Secara umum, ada dua jenis prediksi yaitu kualitatif dan kuantitatif. Prediksi kualitatif merupakan prediksi yang bersifat subjektif, hal ini karena didasarkan pada pengalaman empiris, intuisi pengambilan keputusan dan emosi manusia.Sedangkan, prediksi kuantitatif merupakan prediksi yang bersifat objektif sebab didasarkan pada data aktual dan diolah menggunakan metode tertentu (Surtiningsih, Furqon, & Adinugroho, 2018).

Jenis sesuai yang di jelaskan oleh para ahli sebelumnya prediksi yang digunakan dalam penelitian adalah prediksi kuantitatif. Dikarenakan data yang digunakan adalah data dari masa lalu berupa angka dengan runtutan waktu.

# DAFTAR PUSTAKA

Aldi, M. W., Jondri, & Aditsania, A. (2018, August). Analisis dan Implementasi Long Short Term Memory Neural Network untuk Prediksi Harga Bitcoin. *e-Proceeding of Engineering, 5*, 3548-3555. Diambil kembali dari https://openlibrarypublications.telkomuniversity.ac.id/index.php/engineering/article/view/6739

Alpaydin, E. (2020). *Introduction to Machine Learning, fourth edition.* United States: MIT Press. Diambil kembali dari https://books.google.co.id/books?id=tZnSDwAAQBAJ

Azizah, L. M., Umayah, S. F., & Fajar, F. (2018, November). Deteksi Kecacatan Permukaan Buah Manggis Menggunakan Metode Deep Learning dengan Konvolusi Multilayer. *SEMESTA TEKNIKA, 21*, 230-236. doi:10.18196/st.212229

Boruah, A., & Barman, P. P. (2018). A RNN based Approach for next word prediction in Assamese Phonetic Transcription. *Procedia Computer Science, 143*, 117-123. doi:10.1016/j.procs.2018.10.359

François, C. (2018). *Deep Learning with Python.* Manning. Diambil kembali dari https://books.google.co.id/books?id=Yo3CAQAACAAJ

Ghosh, A., Bose, S., Maji, G., Debnath, N. C., & Sen, S. (2019, September). Stock Price Prediction Using LSTM on Indian. *Proceedings of 32nd international conference, 63*, 101-110. doi:10.29007/qgcz

Gunova, V. (2021, Juni). IMPLEMENTASI DEEP LEARNING PADA SIMULASI AUTONOMOUS DRIVE MENGGUNAKAN AIRSIM. *Journal of Software Engineering, Information and Communication Technology, 2*, 83-92. Diambil kembali dari https://ejournal.upi.edu/index.php/SEICT/article/view/34674

H. M., G. E., Menon, V. K., & S. K. (2018). NSE Stock Market Prediction Using Deep-Learning Models. *Procedia Computer Science, 132*, 1351-1362. doi:10.1016/j.procs.2018.05.050

Hasanah, F. N., & Untari, R. S. (2020). *BUKU AJAR REKAYASA PERANGKAT LUNAK.* (M. Suryawinata, M. Nashrullah, & A. Y. Prajati, Penyunt.) Sidoarjo, Jawa Timur, Indonesia: UMSIDA PRESS.

Hu, J., Niu, H., Carrasco, J., Lennox, B., & Arvin, F. (2020, October 29). Voronoi-Based Multi-Robot Autonomous Exploration in Unknown Environments via Deep Reinforcement Learning. *IEEE Transactions on Vehicular Technology, 69*(12), 14413-14423. doi:10.1109/TVT.2020.3034800

Ilahiyah, S., & Nilogiri, A. (2018, August 2). Implementasi Deep Learning Pada Identifikasi Jenis Tumbuhan Berdasarkan Citra Daun Menggunakan Convolutional Neural Network. *Jurnal Sistem & Teknologi Informasi Indonesia, 3*, 49-56. doi:10.32528/justindo.v3i2.2254

Ilham, W., & Fajri, N. (2020, May). PENENTUAN JUMLAH PRODUKSI TAHU DENGAN MENGGUNAKAN METODE FUZZY TSUKAMOTO PADA UKM ABADI BERBASIS WEB. *JURNAL DIGIT, 10*(1), 71~82. doi:http://jurnaldigit.org/index.php/DIGIT/article/viewFile/158/116

Manaswi, N. K. (2018). *Deep Learning with Applications Using Python.* Bangalore, Karnataka, India. doi:10.1007/978-1-4842-3516-4

Manu. (2021, January 30). *A simple overview of RNN, LSTM and Attention Mechanism*. Diambil kembali dari Start it up: https://medium.com/swlh/a-simple-overview-of-rnn-lstm-and-attention-mechanism-9e844763d07b

Pardede, J., & Ibrahim, R. G. (2020, December 31). Implementasi Long Short-Term Memory Untuk Identifikasi Berita Hoax Berbahasa Inggris Pada Media Sosial. *J-COSINE, 4*, 179-187. doi:10.29303/jcosine.v4i2.361

Putro, B., Furqon, M. T., & Wijoyo, S. H. (2018, November). Prediksi Jumlah Kebutuhan Pemakaian Air Menggunakan Metode Exponential Smoothing (Studi Kasus : PDAM Kota Malang). *Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer, 2*, 4679-4686. Diambil kembali dari http://j-ptiik.ub.ac.id

Qiu, J., Wang, B., & Zhou, C. (2019, December 13). Forecasting stock prices with long-short term memory neural network based on attention mechanism. (T. Song, Penyunt.) *PLoS ONE*, 1-15. doi:10.1371/journal.pone.0227222

Qomariyah, N., Hamzah, N., & Mustika, W. P. (2018, August). PENERAPAN METODE ARTIFICIAL NEURAL NETWORKUNTUK MENDETEKSI SERANGANJANTUNGDIRS AWAL BROS BEKASI. *INTI NUSA MANDIRI, 13*(1), 27-32. Diambil kembali dari http://ejournal.nusamandiri.ac.id/index.php/inti/article/view/1652/746

Rinaldi, R. (2019, February 26). PENERAPAN UNIFIED MODELLING LANGUAGE (UML) DALAM ANALISIS DAN PERANCANGAN APLIKASI E-LEARNING. *Jurnal SIMTIKA, 2*, 43–50. Diambil kembali dari http://ejournal.undhari.ac.id/index.php/simtika/article/view/15

Satria, B. (2018, December 15). Prediksi Volume Penggunaan Air PDAM Menggunakan Metode Jaringan Syaraf Tiruan Backpropagation. *JURNAL RESTI, 2*(3), 674-684.

Savalia, S., & Emamian, V. (2018, April 28). Cardiac Arrhythmia Classification by Multi-Layer Perceptron and Convolution Neural Networks. *Bioengineering, 5*, 2-10. doi:10.3390/bioengineering5020035

Setiyani, L. (2018). *Rekayasa Perangkat Lunak [Software Engineering].* (L. Setiyani, Penyunt.) Karawang, Jawa Barat, Indonesia: Jatayu Catra Internusa.

Shekar, A. K., S´a, C. R., Ferreira, H., & Soares, C. (2018, April). Building robust prediction models for defective sensor data using Artificial Neural Networks.

Siahaan, M., Jasa, C. H., Anderson, K., Rosiana, M. V., Lim, S., & Yudianto, W. (2020, November). Penerapan Artificial Intelligence (AI)Terhadap Seorang Penyandang Disabilitas Tunanetra. *Journal of Information System and Technology, 1*, 186-193. Diambil kembali dari https://journal.uib.ac.id/index.php/joint/article/view/4322/1122

Sihombing, E. N., & Syaputra, M. Y. (2020, September 23). IMPLEMENTASI PENGGUNAAN KECERDASAN BUATAN DALAM PEMBENTUKAN PERATURAN DAERAH. *JURNAL ILMIAH KEBIJAKAN HUKUM, 4*, 419-434. doi:10.30641/kebijakan.2020.V14.419-434

Sucipto, L., & Syaharuddin, S. (2018, September 2). Konstruksi Forecasting System Multi-Model untuk pemodelan matematika pada peramalan Indeks Pembangunan Manusia Provinsi Nusa Tenggara Barat. *Jurnal Ilmiah Teknologi Sistem Informasi, 4*, 114-123. doi:10.26594/register.v4i2.1263

Surtiningsih, L., Furqon, M. T., & Adinugroho, S. (2018, August 8). Prediksi Jumlah Kunjungan Wisatawan Mancanegara Ke Bali Menggunakan Support Vector Regressiondengan Algoritma Genetika. *Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer, 2*, 2578-2586. Diambil kembali dari https://j-ptiik.ub.ac.id/index.php/j-ptiik/article/view/1857

Tabrani, M., & Pudjiarti, E. (2021, August 2). Pembangunan Perangkat Lunak E-Learning dalam Kegiatan Belajar Mengajar Sekolah Menengah Kejuruan Angkasa Husein. *Jurnal Sains Manajemen Informatika dan Komputer, 20*(2), 130~138. Diambil kembali dari https://ojs.trigunadharma.ac.id/

Wahyudi, A. (2018). Perancangan sistem menggunakan metode sdlc. *J. Din. Inform, 4*(2), 1-7.

Wijaya, R., Masriadi, & Ikhlas, M. (2020, July). IMPLEMENTASI MODEL WATERFALL PADA PENGEMBANGAN SISTEM INFORMASI PERHITUNGAN NILAI MATA PELAJARAN BERBASIS WEB PADA SEKOLAH DASAR AL-AZHAR SYIFA BUDI JATIBENING. *INFORMATION SYSTEM DEVELOPMENT, 5*(2).

Zahara, S., Sugianto, & Ilmiddafiq, M. B. (2019, December 2). Prediksi Indeks Harga Konsumen Menggunakan Metode Long Short Term Memory(LSTM) Berbasis Cloud Computing. *JURNAL RESTI (Rekayasa Sistem dan Teknologi Informasi), 3*, 357–363. doi:10.29207/resti.v3i3.1086