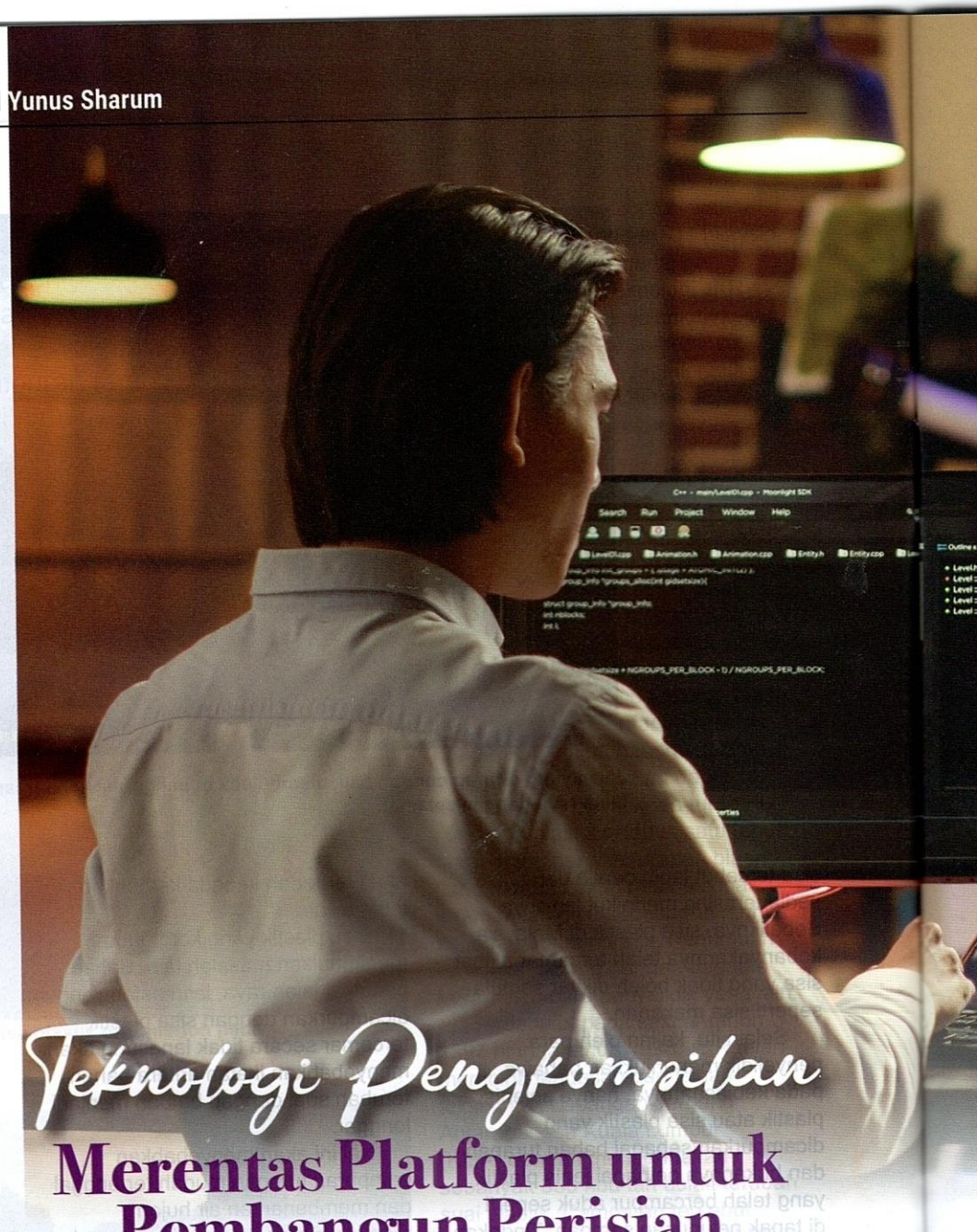


Teknologi Pengkompilan
Merentas Platform untuk Pembangun Perisian

Pengguna komputer menggunakan aplikasi yang diuruskan oleh sistem pengoperasian dan dilaksanakan dalam sebuah komputer yang bersifat pelbagai guna. Dalam konteks keupayaan, komputer pelbagai guna lebih baik berbanding dengan komputer yang bersifat spesifik yang mengawal peranti seperti ketuhar gelombang mikro pintar, televisyen pintar atau yang seumpama dengannya.

Komputer pelbagai guna boleh melaksanakan pelbagai jenis aplikasi pengguna seperti perisian pemproses perkataan, penyunting grafik, pelayar Internet, aplikasi permainan, pemain media dan utiliti. Penggunaan komputer spesifik pula terhad kepada skop fungsinya sahaja.

Pada masa ini, terdapat beberapa jenis komputer pelbagai guna, misalnya komputer peribadi, komputer riba, telefon mudah alih dan tablet. Setiap jenis komputer ini mempunyai pelbagai model yang dihasilkan oleh banyak syarikat pengeluar. Namun demikian, dari perspektif pelaksanaan perisian dan aplikasi, semua komputer tersebut dianggap sama sahaja selagi menggunakan seni bina dan jenis cip pemproses yang sama.

Faktor utama yang membezakan jenis komputer pelbagai guna ialah seni bina cip pemprosesnya. Hal ini disebabkan oleh pemproses yang menggunakan seni bina berbeza akan melaksanakan arahan atur cara dengan cara yang berbeza. Atur cara yang dibina (dikompil) untuk suatu seni bina pemproses hanya boleh dilaksanakan pada komputer dengan pemproses tersebut.

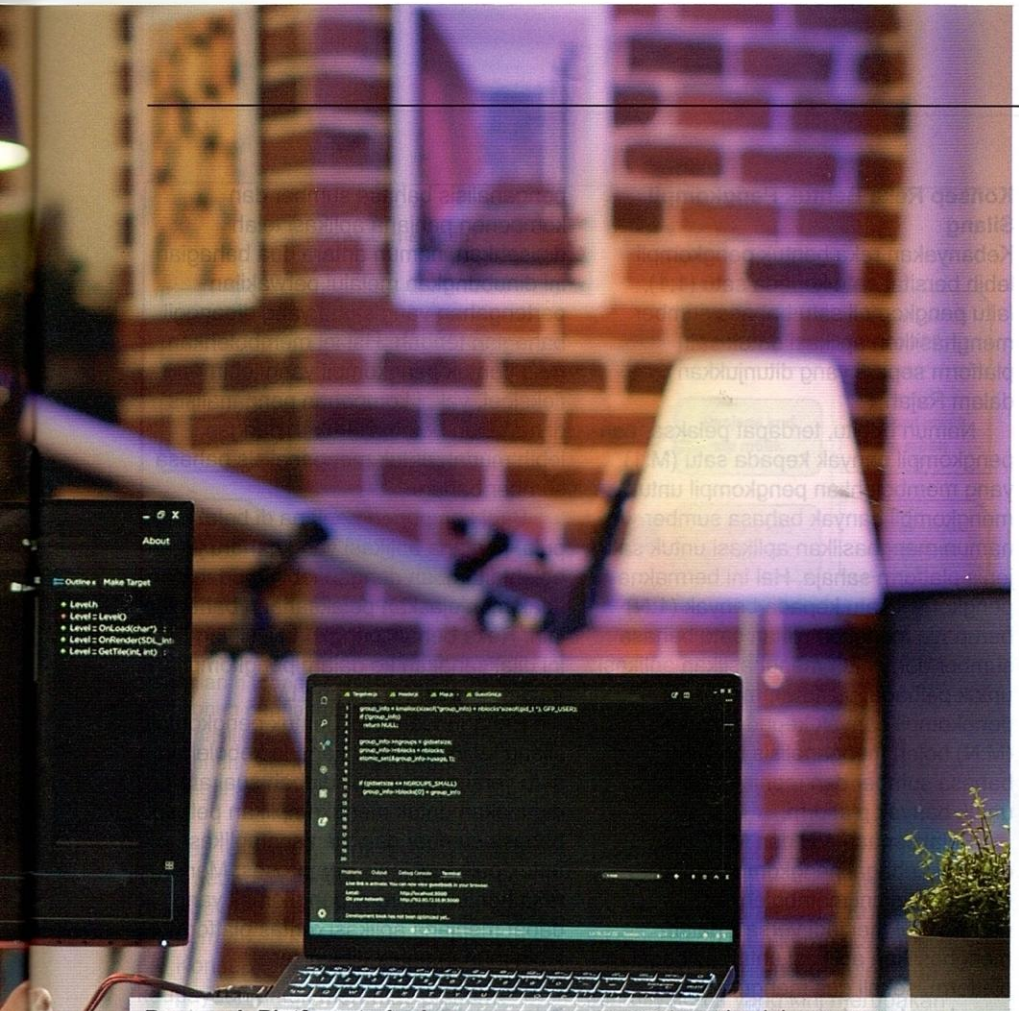
Antara seni bina pemproses yang popular pada masa ini termasuklah x86, m68k, PowerPC dan ARM. Dalam kalangan pemproses tersebut pula terbahagi kepada dua jenis seni bina, iaitu 32-bit dan 64-bit.

Sistem pengoperasian di dalam komputer pula mempunyai pelbagai reka bentuk dan seni bina. Oleh sebab pelaksanaan aplikasi diuruskan oleh berbagai-bagai sistem pengoperasian dengan cara masing-masing, maka aplikasi turut tertakluk kepada jenis sistem pengoperasian yang digunakan.

Antara sistem pengoperasian yang popular pada masa ini termasuklah Windows, Unix, Linux, MacOS, Android dan IOS. Sebahagian daripada sistem ini dibina untuk seni bina pemproses tertentu dan terdapat juga yang mempunyai versi untuk pelbagai pemproses.

Gabungan jenis komputer dan sistem pengoperasian disebut sebagai platform dan sering berkait rapat antara satu sama lain. Hal ini dikatakan demikian kerana sistem pengoperasian tersebut menguruskan pelaksanaan aplikasi secara langsung dengan pemproses. Sebagai contohnya, sistem pengoperasian berfungsi menjadualkan pelaksanaan atau menguruskan capaian ruang ingatan serta capaian peranti input dan output.

Penghasilan atur cara pada awalnya menggunakan bahasa mesin. Bahasa mesin ialah arahan yang khusus untuk seni bina cip pemproses tertentu. Apabila bahasa himpunan diperkenalkan untuk membina atur cara, bentuk arahan bersifat lebih umum, namun masih terikat dengan ciri pemproses sebagai platformnya.



Pengaruh Platform terhadap Pelaksanaan Aplikasi

Aplikasi yang terikat dengan suatu platform mempunyai kelebihan yang tersendiri. Aplikasi sebegini biasanya dapat menggunakan kelebihan spesifik yang terdapat pada platform tersebut.

Selain itu, pelaksanaan aplikasi menjadi lebih pantas kerana disesuaikan mengikut keupayaan platform. Secara umumnya, aplikasi yang terikat dengan platform banyak tertumpu kepada kepantasan pemprosesan. Pengguna juga dapat memanfaatkan ciri khas yang dimiliki oleh platform berkenaan.

Walau bagaimanapun, aplikasi yang terikat dengan platform tertentu boleh menimbulkan masalah terhadap pembina aplikasi. Hal ini disebabkan oleh terdapat lebih daripada satu platform yang digunakan dalam kalangan pengguna. Pembina aplikasi perlu menentukan platform sasaran mereka dengan teliti agar penggunaan aplikasi mencakupi banyak pengguna.

Pada masa yang sama, pembina aplikasi tidak dapat memastikan secara tepat jumlah pengguna yang menggunakan setiap platform. Sekiranya tersilap memilih platform, maka kuranglah pengguna yang akan menggunakan aplikasi tersebut. Pengguna yang berminat tetapi tidak mempunyai platform yang diperlukan turut tidak berpeluang untuk menggunakan aplikasi yang dibina.

Disebabkan oleh faktor tersebut, sebahagian daripada pembina, khususnya yang menjual perisian perlu menyediakan lebih daripada satu versi aplikasi untuk platform yang berbeza. Melalui penghasilan atur cara dengan menggunakan bahasa paras tinggi, perisian atau alat seperti pengkompil dapat membantu menjana aplikasi dengan versi berbeza mengikut platform yang diinginkan.

Namun demikian, situasi tersebut masih boleh mendatangkan masalah tertentu seperti atur cara yang sudah dihasilkan bagi suatu platform tidak dapat dikompil untuk platform yang lain. Jika keadaan ini berlaku, atur cara tersebut terpaksa diubah suai untuk menghasilkan versi baharu yang dapat dilaksanakan pada platform yang gagal sebelumnya. Tindakan ini sekali gus mewujudkan dua atau lebih versi atur cara.

Masalah lain juga boleh berlaku seperti aplikasi yang sudah digunakan masih mengandungi ralat (pepijat) dan atur caranya perlu diperbaik. Dalam hal ini, versi atur cara untuk semua platform perlu diselenggara dengan teliti.

Terdapat kemungkinan bahawa selepas atur cara diperbaik, aplikasi untuk satu platform masih bermasalah sedangkan versi untuk platform lain sudah boleh digunakan. Oleh sebab itulah, banyak atur cara pelbagai versi akan wujud.

Pembina aplikasi juga perlu sentiasa cakna terhadap perkembangan setiap platform kerana sistem pengoperasian juga merupakan perisian yang sentiasa perlu ditambah baik dari segi ciri baharu, kecacatan, pepijat dan sebagainya. Cip pemproses juga perlu dipertingkatkan dan berkemungkinan mengalami perubahan pada seni binanya.

Sebagai contohnya, aplikasi yang dibina untuk seni bina 32-bit sebaik-baiknya dikompil semula jika terdapat seni bina 64-bit. Hal ini perlu supaya aplikasi dapat mengadaptasi kelebihan baharu dan mengetepikan ciri lama yang sudah tiada. Jika tidak, perubahan pada platform boleh menyebabkan aplikasi yang sedia ada tidak dapat memanfaatkan ciri baharu atau mengalami masalah tertentu.

Kewujudan pelbagai platform untuk aplikasi boleh menjadi isu kepada pembina, terutamanya dari aspek penyelenggaraan berbilang versi atur cara dan aplikasi. Maka itu, sesebuah atur cara atau aplikasi perlu memiliki sifat rentas platform.

Selain memudahkan penyelenggaraan dengan satu versi, pembina juga mampu memastikan lebih banyak pengguna dapat menggunakan aplikasi yang dihasilkan.

Teknologi Atur Cara Terkompil

Keupayaan merentas platform pada peringkat atur cara mula dicapai apabila bahasa pengaturcaraan paras tinggi diperkenalkan. Antara

bahasa pengaturcaraan yang terawal termasuklah ADA, FORTRAN, COBOL dan C.

Melalui penggunaan bahasa paras tinggi, kod atur cara yang sama boleh menghasilkan pelbagai versi untuk platform berbeza. Kaedahnya adalah dengan mengkompilkan atur cara tersebut untuk setiap platform sasaran.

Aplikasi untuk platform berkenaan akan terhasil dalam bentuk fail terkompil atau bahasa mesin dan kemudiannya bersedia untuk dilaksanakan. Atur cara asal juga dikenali sebagai bahasa sumber dan kod sumber.

Keupayaan tersebut dicapai kerana bahasa paras tinggi tidak mengkhusus kepada seni bina pemproses, sebaliknya mengikut sintaks bahasanya sendiri. Cara ini hanya memerlukan versi pengkompil berbeza dihasilkan untuk setiap platform. Disebabkan oleh semua komputer telah menggunakan konsep von Neumann, maka keadaan ini memudahkan penghasilan versi pengkompil bahasa untuk platform berbeza.

Namun demikian, aplikasi terkompil untuk suatu platform masih lagi terhad untuk digunakan pada platform itu sahaja. Oleh itu, pembangun aplikasi perlu menentukan platform yang hendak disasarkan sebagai persekitaran pelaksanaan aplikasi.

Konsep Reka Bentuk Pengkompil Silang

Kebanyakan reka bentuk pengkompil lebih bersifat satu kepada satu (1:1), iaitu pengkompil satu bahasa sumber menghasilkan aplikasi untuk satu platform seperti yang ditunjukkan dalam Rajah 1.

Namun begitu, terdapat pelaksanaan pengkompil banyak kepada satu (M:1) yang membolehkan pengkompil untuk mengkompil banyak bahasa sumber, namun menghasilkan aplikasi untuk satu jenis platform sahaja. Hal ini bermakna pengkompil satu kepada banyak (1:N) berupaya mengkompilkan satu bahasa sumber dan boleh menghasilkan aplikasi untuk pelbagai plafform, manakala pengkompil banyak kepada banyak (M:N) boleh menganalisis banyak bahasa sumber dan menjana aplikasi untuk banyak platform.

Pelaksanaan 1:N dan M:N juga dikenali sebagai pengkompil silang. Reka bentuk ini membolehkan pengkompil menjadi versatil dalam menghasilkan aplikasi untuk pelbagai platform, sekali gus mengurangkan keperluan kepada banyak versi pengkompil. Namun begitu, pelaksanaan pengkompil menjadi sukar kerana reka bentuk pengkompil menjadi kompleks dan tidak mudah diselenggara.

Untuk menjadikan reka bentuk pengkompil silang lebih baik, komponen penganalisis bahasa sumber dan komponen penjana aplikasi telah dipisahkan, namun antara dua bahagian ini dihubungkan melalui perwakilan pertengahan yang sama yang dipanggil bahagian tengah. Hal ini membolehkan reka bentuk pengkompil yang lebih fleksibel, yakni komponen penganalisis di hadapan boleh ditambah dengan lebih mudah untuk menganalisis bahasa sumber baharu.

Bagi komponen penjana di belakang pula, penjana aplikasi yang baharu juga boleh ditambah dengan lebih mudah dan tidak menjejaskan pelaksanaan bahagian lain. Reka bentuk pengkompil silang ini dilaksanakan ke dalam pengkompil moden seperti GCC dan LLVM.

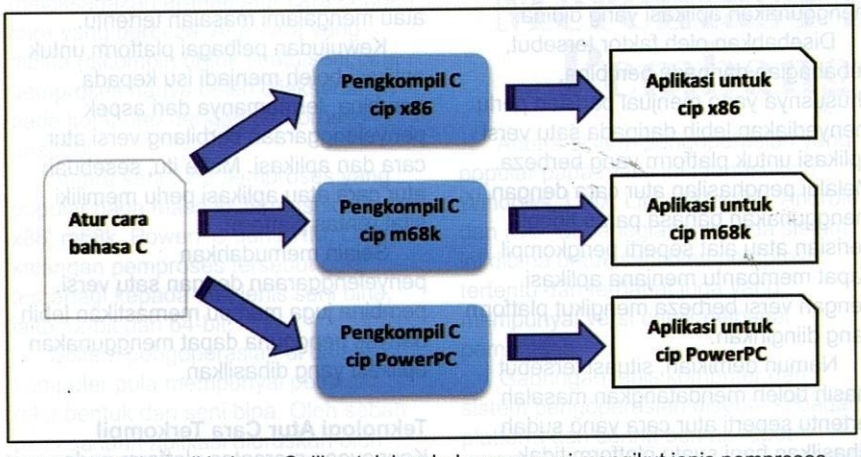
Penggunaan reka bentuk pengkompil silang lebih tertumpu kepada kebolehan satu versi pengkompil diubah suai atau digunakan untuk mengkompilkan pelbagai bahasa sumber dan menjana aplikasi untuk pelbagai jenis pemproses. Menerusi reka bentuk ini, versi pengkompil yang dapat menjana aplikasi rentas platform lebih mudah untuk dihasilkan.

Teknologi Terjemahan Skrip

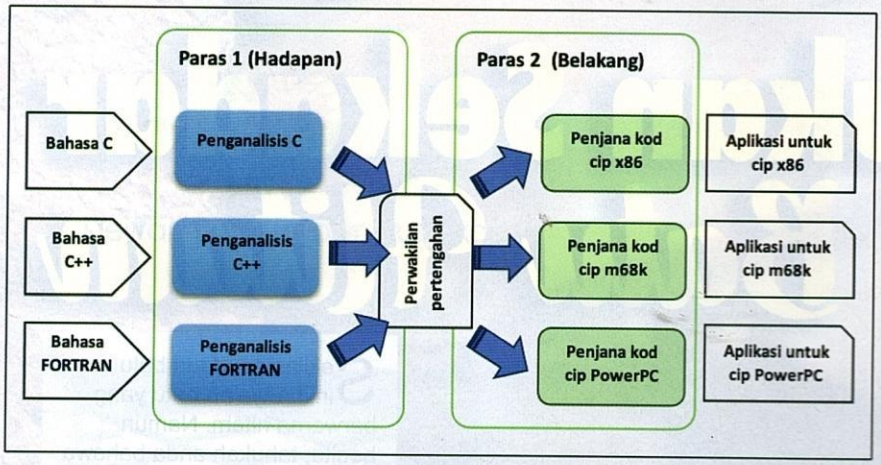
Bagi menjadikan suatu atur cara lebih fleksibel untuk merentas platform, maka pereka bentuk pengkompil telah memperkenalkan konsep bahasa terjemahan. Perubahan konsep terkompil kepada terjemahan hanya membabitkan perubahan kepada proses pengkompilan.

Dalam konsep terjemahan, suatu atur cara dalam bahasa paras tinggi yang dipanggil skrip, dikompilkan semasa hendak dilaksanakan. Hal ini bermakna atur cara aplikasi akan sentiasa disimpan dalam bentuk skrip. Atur cara skrip yang dipanggil penterjemah akan diproses oleh sejenis pengkompil apabila diminta dilaksanakan oleh pengguna. Penterjemah akan memproses arahan atur cara, menukarkannya kepada bahasa mesin atau bahasa perantara dan kemudiannya meminta sistem untuk melaksanakan arahan tersebut.

Konsep ini digunakan oleh bahasa pengaturcaraan JavaScript, Perl, Python, Go, Lua, Tcl dan sebagainya. Terdapat juga sistem yang mengubah pengkompil bahasa yang asalnya menggunakan



Rajah 1 Pengkompil bahasa C dibuat dalam beberapa versi mengikut jenis pemproses yang membolehkan satu atur cara C dikompil ke dalam pelbagai versi mengikut jenis pemproses. Atur cara disimpan dalam bentuk aplikasi bahasa mesin untuk dilaksanakan apabila diperlukan.



Rajah 2 Pengkompil yang menggunakan reka bentuk pengkompil silang memisahkan bahagian hadapan dan belakang. Sebagai contohnya, di bahagian hadapan mengandungi komponen penganalisis bahasa C, C++ dan FORTRAN yang kesemuanya akan menghasilkan satu perwakilan pertengahan. Di belakang pula, terdapat beberapa komponen penjana aplikasi untuk pemproses yang berbeza yang menggunakan perwakilan pertengahan untuk menjana aplikasi bagi platform tertentu.

konsep terkompil, namun menyediakan versi sebagai penterjemah dan membolehkan bahasa itu dikompil atau diterjemahkan kepada sistem tersebut.

Konsep terjemahan membentuk keupayaan untuk merentas platform yang lebih baik. Hal ini adalah kerana skrip tidak perlu dikompilkan terlebih dahulu. Oleh sebab itu, atur cara skrip boleh dilaksanakan pada mana-mana platform dengan kehadiran penterjemah. Perkara yang diperlukan ialah pengguna platform memasang penterjemah bahasa skrip agar boleh melaksanakan skrip sebagai aplikasi.

Teknologi Terjemahan JIT

Konsep terjemahan juga merangkumi konsep *just-in-time* (JIT). Konsep ini digunakan oleh kerangka kerja Java yang diperkenalkan oleh Sun Microsystems yang kini merupakan anak syarikat Oracle Corporation dan NET yang diperkenalkan oleh Microsoft Corporation.

Kedua-dua kerangka kerja ini menyediakan mesin maya untuk menyokong pelaksanaan konsep JIT. Java menggunakan bahasa Java sebagai bahasa sumber, manakala NET menggunakan pendekatan dengan menyokong beberapa jenis bahasa pengaturcaraan sebagai bahasa sumber yang merangkumi VisualBasic. NET, C#, F# dan ASP.NET.

Konsep terjemahan JIT menggunakan pendekatan pertengahan antara konsep terkompil dengan terjemahan, iaitu atur cara bahasa sumber akan dikompilkan ke dalam bentuk kod bait. Konsep JIT juga ada kalanya digolongkan sebagai atur cara terkompil seperti C dan C++.

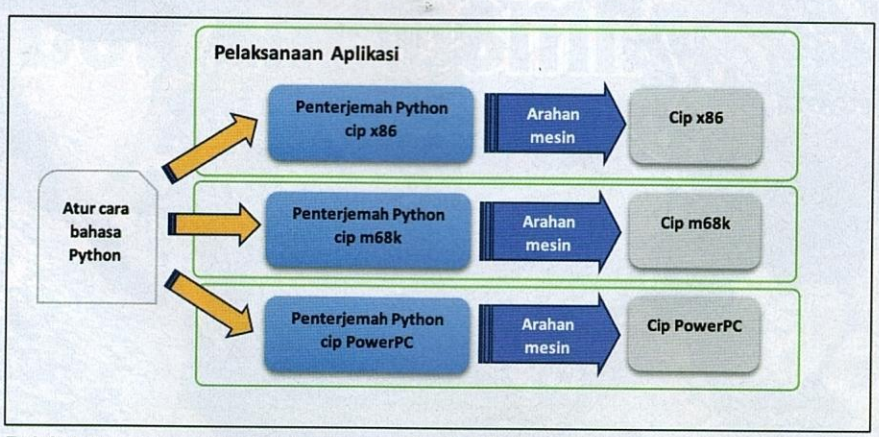
Terjemahan dilakukan ketika aplikasi hendak dilaksanakan. Mesin maya akan "menterjemahkan" kod bait kepada bahasa mesin dan membolehkan aplikasi dilaksanakan dalam platform sasaran.

Pelaksanaan aplikasi menggunakan JIT adalah lebih pantas berbanding dengan terjemahan daripada bahasa skrip. Hal ini berlaku kerana bentuk dan struktur kod bait hampir menyerupai bahasa mesin yang sebenar tetapi masih mengikut formatnya.

Penggunaan aplikasi dalam bentuk kod bait memberikan keupayaan kepada aplikasi untuk merentas platform. Hal ini bermakna bahawa semua aplikasi kod bait boleh dilaksanakan pada mana-mana platform yang sudah dipasang dengan mesin maya JIT tanpa perlu mengubah atur cara kod bait tersebut.

Teknologi pengkompilan merentas platform memberikan kelebihan kepada pembinaan dan pelaksanaan aplikasi. Selain mengurangkan masalah penyelenggaraan, keupayaan ini membolehkan aplikasi yang sama digunakan pada semua platform sasaran.

Beberapa teknologi bahasa dan pengkompilan juga telah direka bagi mengatasi kekangan platform yang merangkumi penggunaan bahasa paras tinggi, reka bentuk pengkompil silang, konsep terjemahan dan terjemahan JIT.



Rajah 3 Atur cara sentiasa disimpan dalam bentuk bahasa Python dan diterjemahkan semasa pelaksanaan aplikasi untuk pemproses berbeza.

Dr. Mohd Yunus Sharum,
Jabatan Sains Komputer,
Fakulti Sains Komputer dan
Teknologi Maklumat,
Universiti Putra Malaysia.