PYTHON QUESTIONS:

1]who is the founder of python?

ANS:

1. [Guido van Rossum](https://www.google.com/search?rlz=1C1CHBF_enIN816IN816&sxsrf=ALeKk01TKbHhv-Bf5Y3MQcHctJS3M1u-GA:1594467325212&q=Guido+van+Rossum&stick=H4sIAAAAAAAAAONgVuLUz9U3MMwwME1-xGjCLfDyxz1hKe1Ja05eY1Tl4grOyC93zSvJLKkUEudig7J4pbi5ELp4FrEKuJdmpuQrlCXmKQTlFxeX5gIACMWaE1cAAAA)
2. Python (programming language) Python is an interpreted, high-level, general-purpose programming language. Created by **Guido van Rossum** and first released in 1991, Python's design philosophy emphasizes code readability with its notable use of significant whitespace.

2] Name some commonly used built-in modules in Python?

1) print() and input() for I/O

2) number conversion functions int() , float() , complex()

3) data type conversions **list**() , tuple() , set()

4) Python Standard Library: Core Modules 1-1 Core Modules "Since the functions in the C runtime library are not part of the Win32 API, we believe the number of applications that will be affected by this bug to be very limited" Microsoft, January 1999 Overview Python's standard library covers a wide range of modules.

Everything from modules that are as much a part of the Python language as the types and statements defined by the language specification, to obscure modules that are probably useful only to a small number of programs.

This section describes a number of fundamental standard library modules. Any larger Python program is likely to use most of these modules, either directly or indirectly. Built-in Functions and Exceptions Two modules are even more basic than all other modules combined: the \_\_builtin\_\_ module defines built-in functions (like len, int, and range), and the exceptions module defines all built-in exceptions.

Python imports both modules when it starts up, and makes their content available for all programs. Operating System Interface Modules There are a number of modules providing platform-independent interfaces to the underlying operating system.

They are modeled after the POSIX standard API and the standard C library. The modules in this group include os, which provides file and process operations, os.path which offers a platform-independent way to pull apart and put together file names, and time which provides functions to work with dates and times. To some extent, networking and thread support modules could also belong in this group, but they are not supported by all Python implementations. Python Standard Library Copyright (c) 1999-2003 by Fredrik Lundh.

All rights reserved. Python Standard Library: Core Modules 1-2 Type Support Modules Several built-in types have support modules in the standard library. The string module implements commonly used string operations, the math module provides math operations and constants, and the cmath module does the same for complex numbers. Regular Expressions The re module provides regular expressions support for Python. Regular expressions are string patterns written in a special syntax, which can be used to match strings, and extract substrings.

Language Support Modules sys gives you access to various interpreter variables, such as the module search path, and the interpreter version. operator provides functional equivalents to many built-in operators. copy allows you to copy objects. And finally, gc gives you more control over the garbage collector facilities in Python 2.0. Python Standard Library: Core Modules 1-3 The \_\_builtin\_\_ module This module contains built-in functions which are automatically available in all Python modules. You usually don't have to import this module; Python does that for you when necessary. Calling a function with arguments from a tuple or dictionary Python allows you to build function argument lists on the fly.

Just put all the arguments in a tuple, and call the built-in apply function: Example: Using the apply function # File:builtin-apply-example-1.py def function(a, b): print a, b apply(function, ("whither", "canada?")) apply(function, (1, 2 + 3)) whither canada? 1 5 To pass keyword arguments to a function, you can use a dictionary as the third argument to apply: Example: Using the apply function to pass keyword arguments # File:builtin-apply-example-2.py def function(a, b): print a, b apply(function, ("crunchy", "frog")) apply(function, ("crunchy",), {"b": "frog"}) apply(function, (), {"a": "crunchy", "b": "frog"}) crunchy frog crunchy frog crunchy frog Python Standard Library: Core Modules 1-4 One common use for apply is to pass constructor arguments from a subclass on to the base class, especially if the constructor takes a lot of arguments. Example: Using the apply function to call base class constructors # File:builtin-apply-example-3.py class Rectangle: def \_\_init\_\_(self, color="white", width=10, height=10): print "create a", color, self, "sized", width, "x", height class RoundedRectangle(Rectangle): def \_\_init\_\_(self, \*\*kw): apply(Rectangle.\_\_init\_\_, (self,), kw) rect = Rectangle(color="green", height=100, width=100) rect = RoundedRectangle(color="blue", height=20) create a green sized 100 x 100 create a blue sized 10 x 20 Python 2.0 provides an alternate syntax. Instead of apply, you can use an ordinary function call, and use \* to mark the tuple, and \*\* to mark the dictionary. The following two statements are equivalent: result = function(\*args, \*\*kwargs) result = apply(function, args, kwargs) Loading and reloading modules If you've written a Python program larger than just a few lines, you know that the import statement is used to import external modules (you can also use the from-import version).

What you might not know already is that import delegates the actual work to a built-in function called \_\_import\_\_. Python Standard Library: Core Modules 1-5 The trick is that you can actually call this function directly. This can be handy if you have the module name in a string variable, like in the following example, which imports all modules whose names end with "-plugin": Example: Using the \_\_import\_\_ function to load named modules # File:builtin-import-example-1.py import glob, os modules = [] for module\_file in glob.glob("\*-plugin.py"): try: module\_name, ext = os.path.splitext(os.path.basename(module\_file)) module = \_\_import\_\_(module\_name) modules.append(module) except ImportError: pass # ignore broken modules # say hello to all modules for module in modules: module.hello() example-plugin says hello Note that the plugin modules have hyphens in the name.

This means that you cannot import such a module using the ordinary import command, since you cannot have hyphens in Python identifiers. Here's the plugin used in this example: Example: A sample plugin # File:example-plugin.py def hello(): print "example-plugin says hello" The following example shows how to get a function object, given that you have the module and function name as strings: Example: Using the \_\_import\_\_ function to get a named function # File:builtin-import-example-2.py def getfunctionbyname(module\_name, function\_name): module = \_\_import\_\_(module\_name) return getattr(module, function\_name) print repr(getfunctionbyname("dumbdbm", "open")) Python Standard Library: Core Modules 1-6 You can also use this function to implement lazy loading of modules. In the following example, the string module is imported when it is first used: Example: Using the \_\_import\_\_ function to implement lazy import # File:builtin-import-example-3.py class LazyImport: def \_\_init\_\_(self, module\_name): self.module\_name = module\_name self.module = None def \_\_getattr\_\_(self, name): if self.module is None: self.module = \_\_import\_\_(self.module\_name) return getattr(self.module, name) string = LazyImport("string") print string.lowercase abcdefghijklmnopqrstuvwxyz Python provides some basic support for reloading modules that you've already imported. The following example loads the hello.py file three times: Example: Using the reload function # File:builtin-reload-example-1.py import hello reload(hello) reload(hello) hello again, and welcome to the show hello again, and welcome to the show hello again, and welcome to the show reload uses the module name associated with the module object, not the variable name.

This means that even if you've renamed the module, reload will still be able to find the original module. Note that when you reload a module, it is recompiled, and the new module replaces the old one in the module dictionary. However, if you have created instances of classes defined in that module, those instances will still use the old implementation.

Likewise, if you've used from-import to create references to module members in other modules, those references will not be updated. Python Standard Library: Core Modules 1-7 Looking in namespaces The dir function returns a list of all members of a given module, class, instance, or other type. It's probably most useful when you're working with an interactive Python interpreter, but can also come in handy in other situations. Example: Using the dir function # File:builtin-dir-example-1.py def dump(value): print value, "=>", dir(value) import sys dump(0) dump(1.0) dump(0.0j) # complex number dump([]) # list dump({}) # dictionary dump("string") dump(len) # function dump(sys) # module 0 => [] 1.0 => [] 0j => ['conjugate', 'imag', 'real'] [] => ['append', 'count', 'extend', 'index', 'insert', 'pop', 'remove', 'reverse', 'sort'] {} => ['clear', 'copy', 'get', 'has\_key', 'items', 'keys', 'update', 'values'] string => [] => ['\_\_doc\_\_', '\_\_name\_\_', '\_\_self\_\_'] => ['\_\_doc\_\_', '\_\_name\_\_', '\_\_stderr\_\_', '\_\_stdin\_\_', '\_\_stdout\_\_', 'argv', 'builtin\_module\_names', 'copyright', 'dllhandle', 'exc\_info', 'exc\_type', 'exec\_prefix', 'executable', ... In the following example, the getmember function returns all class-level attributes and methods defined by a given class: Example: Using the dir function to find all members of a class # File:builtin-dir-example-2.py class A: def a(self): pass def b(self): pass Python Standard Library: Core Modules 1-8 class B(A): def c(self): pass def d(self): pass def getmembers(klass, members=None): # get a list of all class members, ordered by class if members is None: members = [] for k in klass.\_\_bases\_\_: getmembers(k, members) for m in dir(klass): if m not in members: members.append(m) return members print getmembers(A) print getmembers(B) print getmembers(IOError) ['\_\_doc\_\_', '\_\_module\_\_', 'a', 'b'] ['\_\_doc\_\_', '\_\_module\_\_', 'a', 'b', 'c', 'd'] ['\_\_doc\_\_', '\_\_getitem\_\_', '\_\_init\_\_', '\_\_module\_\_', '\_\_str\_\_'] Note that the getmembers function returns an ordered list. The earlier a name appears in the list, the higher up in the class hierarchy it's defined. If order doesn't matter, you can use a dictionary to collect the names instead of a list. The vars function is similar, but it returns a dictionary containing the current value for each member. If you use it without an argument, it returns a dictionary containing what's visible in the current local namespace: Example: Using the vars function # File:builtin-vars-example-1.py book = "library2" pages = 250 scripts = 350 print "the %(book)s book contains more than %(scripts)s scripts" % vars() the library book contains more than 350 scripts Python Standard Library: Core Modules 1-9 Checking an object's type Python is a dynamically typed language, which means that a given variable can be bound to values of different types at different occasions. In the following example, the same function is called with an integer, a floating point value, and a string: def function(value): print value function(1) function(1.0) function("one") The type function allows you to check what type a variable has. This function returns a type descriptor, which is a unique object for each type provided by the Python interpreter. Example: Using the type function # File:builtin-type-example-1.py def dump(value): print type(value), value dump(1) dump(1.0) dump("one") 1 1.0 one Each type has a single corresponding type object, which means that you can use the is operator (object identity) to do type testing: Example: Using the type function to distinguish between file names and file objects # File:builtin-type-example-2.py def load(file): if isinstance(file, type("")): file = open(file, "rb") return file.read() print len(load("samples/sample.jpg")), "bytes" print len(load(open("samples/sample.jpg", "rb"))), "bytes" 4672 bytes 4672 bytes Python Standard Library: Core Modules 1-10 The callable function checks if an object can be called (either directly or via apply). It returns true for functions, methods, lambda expressions, classes, and class instances which define the \_\_call\_\_ method. Example: Using the callable function # File:builtin-callable-example-1.py def dump(function): if callable(function): print function, "is callable" else: print function, "is \*not\* callable" class A: def method(self, value): return value class B(A): def \_\_call\_\_(self, value): return value a = A() b = B() dump(0) # simple objects dump("string") dump(callable) dump(dump) # function dump(A) # classes dump(B) dump(B.method) dump(a) # instances dump(b) dump(b.method) 0 is \*not\* callable string is \*not\* callable is callable is callable A is callable B is callable is callable is \*not\* callable **is callable is callable Note that the class objects (A and B) are both callable; if you call them, they create new objects. However, instances of class A are not callable, since that class doesn't have a \_\_call\_\_ method. You'll find functions to check if an object is of any of the built-in number, sequence, or dictionary types in the operator module. However, since it's easy to create a class that implements e.g. the basic sequence methods, it's usually a bad idea to use explicit type testing on such objects. Python Standard Library: Core Modules 1-11 Things get even more complicated when it comes to classes and instances. Python doesn't treat classes as types per se. Instead, all classes belong to a special class type, and all class instances belong to a special instance type. This means that you cannot use type to test if an instance belongs to a given class; all instances have the same type! To solve this, you can use the isinstance function, which checks if an object is an instance of a given class (or of a subclass to it). Example: Using the isinstance function # File:builtin-isinstance-example-1.py class A: pass class B: pass class C(A): pass class D(A, B): pass def dump(object): print object, "=>", if isinstance(object, A): print "A", if isinstance(object, B): print "B", if isinstance(object, C): print "C", if isinstance(object, D): print "D", print a = A() b = B() c = C() d = D() dump(a) dump(b) dump(c) dump(d) dump(0) dump("string") => A => B => A C => A B D 0 => string => Python Standard Library: Core Modules 1-12 The issubclass function is similar, but checks whether a class object is the same as a given class, or is a subclass of it. Note that while isinstance accepts any kind of object, the issubclass function raises a TypeError exception if you use it on something that is not a class object. Example: Using the issubclass function # File:builtin-issubclass-example-1.py class A: pass class B: pass class C(A): pass class D(A, B): pass def dump(object): print object, "=>", if issubclass(object, A): print "A", if issubclass(object, B): print "B", if issubclass(object, C): print "C", if issubclass(object, D): print "D", print dump(A) dump(B) dump(C) dump(D) dump(0) dump("string") A => A B => B C => A C D => A B D 0 => Traceback (innermost last): File "builtin-issubclass-example-1.py", line 29, in ? File "builtin-issubclass-example-1.py", line 15, in dump TypeError: arguments must be classes Python Standard Library: Core Modules 1-13 Evaluating Python expressions Python provides several ways to interact with the interpreter from within a program. For example, the eval function evaluates a string as if it were a Python expression. You can pass it a literal, simple expressions, or even use built-in functions: Example: Using the eval function # File:builtin-eval-example-1.py def dump(expression): result = eval(expression) print expression, "=>", result, type(result) dump("1") dump("1.0") dump("'string'") dump("1.0 + 2.0") dump("'\*' \* 10") dump("len('world')") 1 => 1 1.0 => 1.0 'string' => string 1.0 + 2.0 => 3.0 '\*' \* 10 => \*\*\*\*\*\*\*\*\*\* len('world') => 5 A problem with eval is that if you cannot trust the source from which you got the string, you may get into trouble. For example, someone might use the built-in \_\_import\_\_ function to load the os module, and then remove files on your disk: Example: Using the eval function to execute arbitrary commands # File:builtin-eval-example-2.py print eval("\_\_import\_\_('os').getcwd()") print eval("\_\_import\_\_('os').remove('file')") /home/fredrik/librarybook Traceback (innermost last): File "builtin-eval-example-2", line 2, in ? File "", line 0, in ? os.error: (2, 'No such file or directory') Note that you get an os.error exception, which means that Python actually tried to remove the file! Python Standard Library: Core Modules 1-14 Luckily, there's a way around this problem. You can pass a second argument to eval, which should contain a dictionary defining the namespace in which the expression is evaluated. Let's pass in an empty namespace: >>> print eval("\_\_import\_\_('os').remove('file')", {}) Traceback (innermost last): File "", line 1, in ? File "", line 0, in ? os.error: (2, 'No such file or directory') Hmm. We still end up with an os.error exception. The reason for this is that Python looks in the dictionary before it evaluates the code, and if it doesn't find a variable named \_\_builtins\_\_ in there (note the plural form), it adds one: >>> namespace = {} >>> print eval("\_\_import\_\_('os').remove('file')", namespace) Traceback (innermost last): File "", line 1, in ? File "", line 0, in ? os.error: (2, 'No such file or directory') >>> namespace.keys() ['\_\_builtins\_\_'] If you print the contents of the namespace variable, you'll find that it contains the full set of built-in functions. The solution to this little dilemma isn't far away: since Python doesn't add this item if it is already there, you just have to add a dummy item called \_\_builtins\_\_ to the namespace before calling eval: Example: Safely using the eval function to evaluate arbitrary strings # File:builtin-eval-example-3.py print eval("\_\_import\_\_('os').getcwd()", {}) print eval("\_\_import\_\_('os').remove('file')", {"\_\_builtins\_\_": {}}) /home/fredrik/librarybook Traceback (innermost last): File "builtin-eval-example-3.py", line 2, in ? File "", line 0, in ? NameError: \_\_import\_\_ Note that this doesn't product you from CPU or memory resource attacks (for example, something like eval("'\*'\*1000000\*2\*2\*2\*2\*2\*2\*2\*2\*2") will most likely cause your program to run out of memory after a while) Python Standard Library: Core Modules 1-15 Compiling and executing code The eval function only works for simple expressions. To handle larger blocks of code, use the compile and exec functions: Example: Using the compile function to check syntax # File:builtin-compile-example-1.py NAME = "script.py" BODY = """ prnt 'owl-stretching time' """ try: compile(BODY, NAME, "exec") except SyntaxError, v: print "syntax error:", v, "in", NAME syntax error: invalid syntax in script.py When successful, the compile function returns a code object, which you can execute with the exec statement: Example: Compiling and executing compiled code # File:builtin-compile-example-2.py BODY = """ print 'the ant, an introduction' """ code = compile(BODY, "**

**3]** **What is \_\_init\_\_?**

**Ans-**

1)\_\_**init**\_\_ :

"\_\_init\_\_" is a reseved method in **python** classes. It is known as a constructor in object oriented concepts. This method called when an object is created from the class and it allow the class to **initialize** the attributes of a class.

**2)\_\_init\_\_** is a special Python method that is automatically called when memory is allocated for a new object. The sole purpose of **\_\_init\_\_** is to **initialize** the values of instance members for the new object.

3) The \_\_init\_\_ method is similar to **constructors**in C++ and Java. Constructors are used to initialize the object’s state. The task of constructors is to initialize(assign values) to the data members of the class when an object of class is created. Like methods, a constructor also contains collection of statements(i.e. instructions) that are executed at time of Object creation. It is run as soon as an object of a class is instantiated. The method is useful to do any initialization you want to do with your object.

4) \_\_init\_\_ does act like a constructor. You'll need to pass "self" to any class functions as the first argument if you want them to behave as non-static methods. "self" are instance variables for your class.

5) \_\_init\_\_ is an oop construct. \_\_init\_\_ is the constructor for a class. Just like mentioned above, the \_\_init\_\_ method is called as soon as the memory for the object is allocated.

6) When a class defines an [**\_\_init\_\_()**](https://docs.python.org/2/reference/datamodel.html#object.__init__) method, class instantiation automatically invokes [**\_\_init\_\_()**](https://docs.python.org/2/reference/datamodel.html#object.__init__) for the newly-created class instance.

7) the [**\_\_init\_\_()**](https://docs.python.org/2/reference/datamodel.html#object.__init__) method may have arguments for greater flexibility

8) The \_\_init\_\_ method is run as soon as an object of a class is instantiated. The method is useful to do any initialization you want to do with your object. Notice the double underscore both in the beginning and at the end in the name.

9) Here, we define the \_\_init\_\_ method as taking a parameter name (along with the usual self). Here, we just create a new field also called name. Notice these are two different variables even though they have the same name. The dotted notation allows us to differentiate between them.

Most importantly, notice that we do not explicitly call the \_\_init\_\_ method but pass the arguments in the parentheses following the class name when creating a new instance of the class. This is the special significance of this method

10) The **\_\_init\_\_ function** is called a constructor, or initializer, and is automatically called when you create a new instance of a class. Within that **function**, the newly created object is assigned to the parameter self . The notation self. legs is an attribute called legs of the object in the variable self

NLP(NATURAL LANGUAGE PROCESSING) QUESTIONS

1]What does a NLP pipeline consist of?

What does a NLP pipeline consist of?

Natural Language Processing Pipeline Filip Gralinski, Krzysztof Jassem, Marcin Junczys-Dowmunt ´ Abstract The paper presents the main ideas and the architecture of the open source PSI-Toolkit, a set of linguistic tools being developed within a project financed by the Polish Ministry of Science and Higher Education. The toolkit is intended for experienced language engineers as well as casual users not having any technological background. The former group of users is delivered a set of libraries that may be included in their Perl, Python or Java applications.

The needs of the latter group should be satisfied by a user friendly web interface. The main feature of the toolkit is its data structure, the so-called PSI-lattice that assembles annotations delivered by all PSI tools. This cohesive architecture allows the user to invoke a series of processes with one command. The command has the form of a pipeline of instructions resembling shell command pipelines known from Linux-based systems. 1 Introduction 1.1 PSI-Toolkit – General Outline PSI-Toolkit consists of a set of tools (called processors) that aim at processing natural language texts. There are three types of processors: readers, annotators, and writers

. A reader creates the main data structure, the so-called PSI-lattice (see Section 2.), from an external source of information, e.g. from a file (a text file, HTML file, PDF file) or keyboard. An annotator (e.g. a tokenizer, a lemmatizer, a parser) adds new annotations in the form of new edges to the PSI-lattice. Finally, a writer writes back a PSI-lattice to an output device (e.g a file or screen). Faculty of Mathematics and Computer Science Adam Mickiewicz University ul.

Umultowska 87, 61-614 Poznan, Poland ´ e-mail: {filipg, jassem, junczys}@amu.edu.pl 1 2 Filip Gralinski, Krzysztof Jassem, Marcin Junczys-Dowmunt ´ 1.2 An Overview of NLP Toolkits The first widely used NLP toolkit was GATE (General Architecture for Text Engineering) [1] designed in the mid-nineties and still being developed. Currently, dozens of open-source toolkits are available; Wikipedia lists over 30 of them.1 Most toolkits deal with the English language, but toolkits designed for other languages (like TESLA – German [2], Nooj – French [3], Apertium – Spanish [4]) gain popularity. The situation is a bit different for Slavonic languages. In spite of the rapid progress in the development of linguistic tools and resources for Slavonic languages in the 21st century, attempts at organizing them in the format of cohesive toolkits are still sparse and rare. 1.3 PSI-Toolkit Assumptions 1.3.1 Free Licence PSI-Toolkit is released under Lesser General Public License (LGPL). This is a free license, which, in addition to GPL, allows for linking with proprietary software and further distributing the result under any terms (also for business purposes). 1.3.2 Easy Access via a Web Browser All PSI-Toolkit tools may be accessed and tested on-line via a web browser interface at psi-toolkit.wmi.amu.edu.pl. Figure 1. shows an exemplary usage of the toolkit in a web window. The user inputs a text into an edit box (e.g. Electric Light Orchestra) and specifies a command as a sequence of processors that should be executed on the text (e.g. txt-reader ! tp-tokenizer --lang en ! psi-writer). The processors are run in the order specified in the command (here: read a raw text, tokenize the text according to rules of the English language, write the output in the dedicated PSI format). The PSI output lists every edge of the PSI-lattice (see: 2.1) in a separate line. In Figure 1: line 01: corresponds to the edge spanning over the first 8 characters of the input (start position is equal to 0000, offset is equal to 0008).

The type of the edge is “token”, the value of the token is Electric, the lemma the token belongs to is Electric. Line 02 describes a space between the first and the second token of the input. Line 05 corresponds to the edge spanning over the whole input (the edge has been constructed by the txt-reader). The format of the output may be simplified by replacing the psi-writer with simple-writer in the command line. Combining the simple output format with 1 http://en.wikipedia.org/wiki/List of natural language processing toolkits.

PSI-Toolkit: A Natural Language Processing Pipeline 3 Fig. 1: Web access to PSI-Toolkit. other PSI-Toolkit processors may result, for instance, in on-line machine translation of the input, yielding a use case similar to that offered by Apertium. 1.3.3 Linux Distribution The PSI-Toolkit is also distributed in the form of two Linux binaries: psi-pipe and psi-server. psi-pipe may be installed and used as a command-line tool on personal computers, whereas psi-server allows for the creation of other PSIToolkit web pages. Currently, the PSI-Toolkit binaries are distributed in form of easily installable packages for the Ubuntu Linux distribution. 1.3.4 PSI-Toolkit Pipeline The PSI-Toolkit command is specified as a pipeline of processors (an example of such a pipeline was given in 1.3.2). If the PSI-Toolkit is used under Ubuntu on a personal computer, the processors should be invoked in a bash-like manner. For example, in order to process the string Electric Light Orchestra in a way equivalent to that shown in 1.3.2., the following pipeline should be formed: > echo "Electric Light Orchestra" | psi\_pipe \ txt\_reader ! tp-tokenizer --lang en ! psi-writer \ | grep "T\_" | less -S 4 Filip Gralinski, Krzysztof Jassem, Marcin Junczys-Dowmunt ´ Note the resemblance of the original linux piping syntax by providing processors chained with ! as arguments to the psi-pipe tool. Moreover, PSI processors may be replaced or supplemented by external tools in the PSI pipeline.

The PSI engine will add annotations provided by external tools to the PSI-lattice. It is admissible to use two different processors of the same type in the same pipeline. For example running two different sentence splitters in the same process may result in two different sentence splits. The ambiguity is stored in the PSI-lattice (it may or may not be resolved in further processing). 1.3.5 Java, Perl, Python Libraries PSI tools are also accessible as libraries of selected programming languages. The user may include the PSI tools in any of the Java, Perl or Python source codes. 1.3.6 Switches Usage of a PSI-Toolkit processor may be customized by means of switches (options). Figure 2. shows the list of available switches for the perl-simple-writer processor, the “Perl-wrapped” version of the PSI simple-writer.

Allowed options: --linear skips cross-edges --no-alts skips alternative edges --with-blank does not skip edges with whitespace text --tag arg (=token) basic tag --spec arg specification of higher-order tags --with-args if set, returns text with annotation as a hash element Fig. 2: Switches of the Perl-simple-writer processor 1.3.7 Natural Languages Processed by PSI-Toolkit There are no restrictions on languages analyzed by the PSI processors (UTF-8 is used for internationalization). One PSI pipeline may consist of processors defined for various languages. However, the tools delivered by the authors are oriented mainly towards Polish (some processors are also defined for English). The authors hope that the PSI-Toolkit can bring together most Polish language processing tools in one framework. Currently these tools are dispersed (see http://clip.ipipan.waw.pl for an exhaustive list of NLP tools and resources for Polish). The proof of this concept has been implemented on the morphosyntactic level: the external morphologi- PSI-Toolkit: A Natural Language Processing Pipeline 5 cal tagger Morfologik2 has been incorporated into the PSI-Toolkit and can be run in the PSI pipeline instead of or besides (!) the standard morphological analyzer of the PSI-Toolkit. 2 PSI-lattice 2.1 Definition All PSI-Toolkit processors operate on a common lattice-based data structure called PSI-lattice. The term lattice refers to a (word) lattice [5] as used in natural language processing rather than to a more general notion of abstract algebra. A PSI lattice is composed of an input (substrate) string and edges spanning substrings of the substrate string. Lattice readers read the substrate string (usually from a file) and add some initial edges – usually edges spanning single symbols (a symbol is a character occurring as part of a natural-language text) and edges representing mark-up tags of a given format (e.g. the PSI-Toolkit HTML reader would construct edges encapsulating HTML tags). PSI-Toolkit annotators create new PSI-lattice edges based on existing ones and/or the substrate string, e.g. a tokenizer groups symbol edges into token edges, a lemmatizer creates edges representing lemmas and lexemes for each token edge, a parser produces new parse edges based on the lexeme edges and previously added parse edges. Finally, lattice writers do not add any new edges, they just output all or selected PSI-lattice edges in a required format.

A PSI-lattice edge consists of the following elements: • source and target vertices (PSI-lattice vertices are defined as inter-character points), • annotation item, • layer tags, • partitions, • score (weight). The annotation item conveys the description of the language unit represented by a given edge. An annotation item is realized as an attribute-value matrix in which two attributes are obligatory: category and text. The meaning and interpretation of these two attributes varies between the PSI-Toolkit processors, e.g. the category of a token edge is its type (blank, word, punctuation mark etc.) and its text attribute is just the token itself (as a string), whereas for a lexeme edge the part of speech of the given lexeme is used as the category attribute and its identifier (e.g. long+adj for the adjective long) – as the text attribute. Other attributes are used to describe 2 Available at http://morfologik.blogspot.com. To our knowledge, this tool has not been described in a published scientific work. 6 Filip Gralinski, Krzysztof Jassem, Marcin Junczys-Dowmunt ´ particular features of the given language unit, e.g. morphosyntactic features such as case, gender, tense, person etc. Layer tags are used to express some meta-information associated with a given edge, e.g.: • edge type – whether an edge represents a token, lemma, lexeme, parse etc., • the name of the processor that added the edge, • tagset used in the annotation item. The important point is that edges with the same annotation item are collapsed into a single edge, even if they have different layer tags (with the exception of plane tags – more on this later). In other words, if a processor produces an edge with the same annotation item and the same source and target vertices as some edge already present in the PSI-lattice, then no new edge is added to the PSI-lattice, the two edges are merged instead, the sum of their layer tags is assigned to the updated edge. As it is not always reasonable to collapse two edges with the same annotation item (for instance in the context of machine translation it would not make sense to equate a source-language lexeme and its target-language equivalent when they happen to have the same canonical form and the same attributes), so-called plane (layer) tags are introduced. Plane tags divide a PSI-lattice into a set of disjoint planes, i.e. edges belonging to different PSI-lattice planes (having different plane tags) will not be collapsed even if they share the same annotation item. By convention, plane tags begin with an exclamation mark. Language-code tags specifying the language of a given language unit (e.g. !pl, !en, !de tags) are typical examples of plane tags. When a processor combines some edges into a new edge, the new edge will inherit the plane tags of the subedges by default, unless a list of plane edges was specified explicitly while creating a new edge. A partition specifies which edges were used to create a given edge. For example for a parse edge the partition is a sequence of lexemes (terminals) and subparses (non-terminals) directly combined into the given edge. An edge may have more than one partition, e.g. an edge spanning the expression Electric Light Orchestra may be a parse partitioned into Electric + Light Orchestra or into Electric Light + Orchestra, or it could be a lexeme produced by a multi-word unit lexicon (partitioned into tokens in this interpretation). Each partition is assigned the following properties: layer tags, score and (optionally) rule identifier. Rule identifier is an arbitrary number the interpretation of which varies between the PSI-Toolkit processors, e.g. for a parser it could be an identifier of a grammar rule (a partition could be linked to a rule of the parser’s grammar this way). The score (of an edge or a partition) is a floating point value for which the following properties hold: • the score of an edge is the maximum score of its partitions, • the score of a partition is the sum of scores of its subedges plus some score for the rule that generated the partition. For instance, a score for a parse edge/partition could be interpreted as the log probability of the parse and the score of a parse partition could be calculated as PSI-Toolkit: A Natural Language Processing Pipeline 7 the sum of log probabilities of its subedges and the log probability of the grammar rule applied. PSI-lattice scores, however, does not have to be interpreted as (log) probabilities. They might be treated as some kind of weights or penalties (the latter for negative scores). Taking the maximum score of the partitions (rather than the sum of the partition scores) as the score of an edge might seem controversial from a formal point of view. The reason why we decided to use the maximum value is that the partitions do not have to be independent. For example we would like to run two different lemmatizers on the same words. 2.2 Motivation and Design Assumptions In other toolkits and even more so in the case of independent stand-alone tools the internal data representation differs in general from the input or output data. In consequence, the internal representation is also different between two kinds of tools, e.g. a tokenizer works mostly on raw string data, while a parser works on trees, forests or possibly charts, a statistical machine translation like Moses uses implicit hypothesis graphs etc. During the conversion of the internal representation to a readable output format usually a lot of information is purposefully discarded. This is the case if only the first-best interpretation for an ambiguous problem is provided, for instance the most probable tokenization or the best-scored translation. Alternative interpretations can often only be retrieved by analysing log data, the format of which is most certainly not standardized between different tools or toolkits. Providing alternative interpretations to a follow-up tool involves then a lot of manual intervention. The PSI-lattice is designed to tackle these problems. Firstly, the PSI-lattice is supposed to provide a one-size-fits-all data structure that can be used to contain any annotation generated by various language processing tools without losing information provided by previous processing steps. Secondly, a standardized way to pass around alternate interpretations besides the single-best results is provided. That way, one can easily take advantage of delayed disambiguation, i.e. a higher-order tool can choose among alternatives that a lower-order tool was not able to fully disambiguate. Examples might be a parser that chooses between two given part-ofspeech tags or a syntax-based machine translation application that translates parseforests rather than parse trees. Finally the PSI-lattice obsoletes the conversion from an internal data representation to an external output format as it can itself serve for internal data representation, being perfectly capable of representing alternative tokenizations, parse forests or translation ambiguities. This will be illustrated in the next section where a shallow parser directly constructs a PSI-lattice as output data while using it as an intermediate representation of parses obtained so far. The ease of data interchange between atomic application leads to the second important design guideline of the presented toolkit: Simplified combination of tools. Using available toolkits can be a challenge by itself as their special syntax or con- 8 Filip Gralinski, Krzysztof Jassem, Marcin Junczys-Dowmunt ´ struction can result in quite a steep learning curve. For instance, the NLTK requires at least a basic knowledge of the Python programming language. For the PSIToolkit it has been decided to walk down a path that has already been cut through — in a context much broader than natural language processing. As far as basic text-processing capabilities are concerned, the popular shells in Linux or other Unix-based operation systems can themselves be considered as capable text-processing toolkits. The rationale behind them is to provide a set of small applications or commands where each tool by itself is limited in functionality in the sense that it can perform only exactly one task. However, each tool is supposed to perform particularly well for the task it has been designed for. Typically, data for a tool is provided on its standard input while the results can be read from its standard output. The power of the command-line comes then from the possibility to combine these tools into pipelines by directing standard output from one tool to the standard input of another tool, thus building up a much more sophisticated application from smaller parts. For instance, one can easily create basic sorted frequency lists of words in a text file combining sed, sort, uniq into a pipeline without the need to write specialized scripts in a more complex programming language like Perl. The operations of each tool in a pipeline can be adjusted by switches — for instance the reverse order of sorting for the sort command by sort -r. Due to the popularity of Linux-based systems in the natural language processing community it is safe to assume a high degree of familiarity of the average NLPresearcher with one of the available Linux shells. This assumption leads directly to the second important design decision for our toolkit. Concerning usability it lends itself to exploit this familiarity by simulating the look and feel of the command-line shells in Linux. This includes the construction of a growing set of self-contained single-purpose applications that can be chained into complex pipelines by the use of a familiar looking piping syntax. Also, additional options to any such building-block are provided by switches that again mimic their Linux-shell counterparts. Every pipeline constructed in such a way constitutes a new stand-alone tool, with well-defined input data, output data and functionality. Formally speaking, a pipeline can be seen as self-contained object similar to a functor, the elements of the pipeline being in turn comparable to combinators that make up the functor. While in Linux shells data is typically passed around in text or plain binary format, the exchange format between the PSI-Toolkit components is the described PSI-lattice. 3 Language Processing with PSI-lattice 3.1 Segmentation For both, tokenization and sentence splitting, the rules created for the Translatica machine translation system3 were used (the rules had been published under 3 http://translatica.pl PSI-Toolkit: A Natural Language Processing Pipeline 9 the GNU Lesser General Public Licence). The tokenizer (the processor is called tp-tokenizer) uses a Translatica in-house format for “cutting-off” rules (each rule specifies a regular expression describing a token of a given type, only the first matching rule is applied), whereas the sentence splitter (called srx-segmenter) uses the SRX (Segmentation Rules eXchange) standard4 . It is quite straightforward to store alternative segmentations (on both token and sentence level) in a PSI-lattice and to take them into account in the subsequent processing stages (in lemmatization, parsing etc.). For the time being, both tp-tokenizer and srx-segmenter produce only one segmentation, as it is not possible to express segmentation non-determinism in either the tokenization or SRX rules. Since there are hard-to-disambiguate cases for token/sentence segmentation — e.g. in Polish gen. is either an abbreviation for generał (= general, a military rank) or the word gen (= gene) at the end of a sentence — i.e. cases in which the decision must be postponed to a later processing stage5 , we are considering enhancing the segmentation rules with some non-determinism. In the case of sentence breaking, however, it would involve extending the widely-used SRX standard. For the time being, segmentation ambiguity could be achieved (at least to some extent) with running segmentation processors twice with slightly different set of rules (e.g. once with gen. listed as an abbreviation, once – not listed). Both tp-tokenizer and srx-segmenter can be run with an option specifying the maximum length of, respectively, a token and a sentence. In fact, there exist two types of length limits: a soft one and a hard one – in case of the soft limit a token/sentence break is forced only on spaces, whereas exceeding the hard limit always triggers segmentation. Such limits were introduced for practical reasons as a safeguard against extremely long tokens/sentences which may occur when “garbage” (e.g. unrecognized binary data) is fed to PSI-Toolkit (very long tokens or sentences might slow down the subsequent processing to an unacceptable degree). So far, PSI-Toolkit handles tokenization and sentence breaking for English, French, German, Italian, Polish, Russian and Spanish. 3.2 Lemmatization and Lexica We plan to incorporate as many open source lemmatizers and lexica as possible into PSI-Toolkit. So far, we have created a general framework for adding new lemmatizers into PSI-Toolkit (now it is relatively easy to add a new lemmatizer on condition that a simple function returning all the morphological interpretations is provided by a given lemmatizer) and incorporated the aforementioned Morfologik lemmatizer for Polish. We are in the process of developing our own finite state library (which 4 http://www.gala-global.org/oscarStandards/srx/srx20.html 5 E.g. in named entity recognition or in parsing, see [6] for discussion of tokenization ambiguity 10 Filip Gralinski, Krzysztof Jassem, Marcin Junczys-Dowmunt ´ will be used not only for lemmatization, but also for syntax-based machine translation) and adding support for the Stuttgart Finite State Transducer Tools (SFST) [7]. 3.3 Shallow Parsing The PSI-Toolkit includes a shallow parser – Puddle – that can be used to work with any language as long as a appropriate grammar is provided. It started out as a C++ adaptation of the Spejd [8] shallow parser which was a pure Java tool a that time. By now, Puddle has evolved into an independent tool that has been used as a parser for French, Spanish, and Italian in the syntax-based statistical machine translation application Bonsai [9]. The latest version of Puddle has been redesigned to work with the PSI-lattice as an input and output data structure. The parse tree itself is also constructed directly on top of the input lattice. Consecutive iterations work on a PSI-lattice that has been extended by exactly one edge in the previous iteration. The shallow parsing process of Puddle relies on a set of string matching rules constructed as regular expressions over single characters, words, part-of-speech tags, lemmas and grammatical categories etc. Apart from the matching portion of a rule, it is also possible to define matching patterns for left and right contexts of the main match. The parse tree construction process is linear, matching rules are applied iteratively in a deterministic fashion. The first possible match is chosen and a spanning edge is added to the lattice. No actual search is performed which puts a lot of weight onto the careful design of the parsing grammar. The order of the rules in a grammar determines the choice of rules to be applied to a sentence. The parsing process is finished if no rules can be applied during an iteration. For the parser to work properly on different types of information, the PSI-lattice already needs to contain edges for tokens, lemmas and morphological properties, so it has to be the result of a pipeline that generated this kind of data. The lattice does not need to be previously disambiguated (although that can be helpful and implemented by adding one or more POS-Taggers to the pipeline) since the parser can also work as a disambiguation tool. Matching rules that require morphological agreement between matched symbols (e.g. between a noun and an adjective) can mark edges that contradict this agreement requirement as “discarded”. However, discarded edges are never quite deleted from the PSI-lattice since they can be of importance in later higher-order processing. The parser adds a special type of edge to the lattice marked with “parse” tag. The partition of the edge contains information which subedges have been used to construct the new edge. If there are several possible interpretations all of them are added to the lattice. Syntactic heads are marked with additional tags in their respective edges. PSI-Toolkit: A Natural Language Processing Pipeline 11 4 Work-flow: An End-to-end Example In this section we will illustrate an example pipeline for a short Polish phrase przykładowa analiza (eng. example analysis). Figure 3 describes the stepwise construction of the PSI-lattice for this phrase. The following commands have been used to create the corresponding lattices: a) txt-reader ! dot-writer b) txt-reader ! tp-tokenizer --lang pl ! dot-writer c) txt-reader ! tp-tokenizer --lang pl ! morfologik ! dot-writer d) txt-reader ! tp-tokenizer --lang pl ! morfologik ! puddle --lang pl ! dot-writer The common “txt-reader” is used to convert a text string into an unannotated lattice. The characters of that string are the smallest units in the PSI-lattice (figure 3a). All figures have themselves been generated by another PSI-Toolkit processor, dot-writer, that converts PSI-lattices to graphs described in the DOTlanguage. This can be interpreted for instance by tools from the GraphViz library [10] available at http://graphviz.org. The toolkit contains also a writer named gv-writer that uses the GraphViz library to generated pdf or svg files directly. A Polish language tokenizer (figure 3b) is inserted between the reader and writer processors and adds the first level of annotation to the PSI-lattice. Edges that span characters mark tokens (T) and blanks (B), other symbol types could include punctuation information or HTML mark-up data. Alternative tokenization results could be included in the PSI-lattice as well, this would in most cases results in crossing edges. The third level in the PSI-lattice (figure 3c) is the result of the application of the Morfologik morphological analyzer. Different morphological interpretations are added as individual edges (the morphological features have been omitted in the illustration due to space requirements). The first token przykładowa has several interpretations as an adjective (adj), similarly analiza has been assigned two interpretations as a noun subst. Finally the last level of annotation (figure 3d) is added by the shallow parser described in the previous section. The ambiguity of the morphological annotation results in an ambiguous parsing result with two parallel adjective phrase edges (AP). Together with the following noun this adjective phrases forms a noun phrase (NP) that spans the entire input string. To facilitate the analysis for the shallow parser a part-of-speech tagger can be added between the morphological analyzer and the parser. Currently, a simple maximum entropy-based tagger processor is being added to the toolkit. For the moment, the shallow parser represents the highestorder processor in the PSI-Toolkit, but deep parsers, machine-translation processors and many other tools will be available in the near future, as well as converters for existing tools like the mentioned Morfologik processor.

2] Where can NER be used?

1)

[](https://meta.wikimedia.org/wiki/Special:MyLanguage/Wikipedia_Pages_Wanting_Photos)**[join the WPWP Campaign to help improve Wikipedia articles with photos and win a prize](https://meta.wikimedia.org/wiki/Special:MyLanguage/Wikipedia_Pages_Wanting_Photos)**

[Hide](https://en.wikipedia.org/wiki/Named-entity_recognition)

Named-entity recognition

From Wikipedia, the free encyclopedia

[Jump to navigation](https://en.wikipedia.org/wiki/Named-entity_recognition#mw-head)[Jump to search](https://en.wikipedia.org/wiki/Named-entity_recognition#searchInput)

*"Named entities" redirects here. For HTML, XML, and SGML named entities, see*[*List of XML and HTML character entity references*](https://en.wikipedia.org/wiki/List_of_XML_and_HTML_character_entity_references)*.*

**Named-entity recognition** (**NER**) (also known as **entity identification**, **entity chunking** and **entity extraction**) is a subtask of [information extraction](https://en.wikipedia.org/wiki/Information_extraction) that seeks to locate and classify [named entities](https://en.wikipedia.org/wiki/Named_entity) mentioned in [unstructured text](https://en.wikipedia.org/wiki/Unstructured_data) into pre-defined categories such as person names, organizations, locations, [medical codes](https://en.wikipedia.org/wiki/Medical_classification), time expressions, quantities, monetary values, percentages, etc.

Most research on NER systems has been structured as taking an unannotated block of text, such as this one:

Jim bought 300 shares of Acme Corp. in 2006.

And producing an annotated block of text that highlights the names of entities:

[Jim]Person bought 300 shares of [Acme Corp.]Organization in [2006]Time.

In this example, a person name consisting of one token, a two-token company name and a temporal expression have been detected and classified.

State-of-the-art NER systems for English produce near-human performance. For example, the best system entering [MUC-7](https://en.wikipedia.org/wiki/Message_Understanding_Conference) scored 93.39% of [F-measure](https://en.wikipedia.org/wiki/F1_score) while human annotators scored 97.60% and 96.95%.[[1]](https://en.wikipedia.org/wiki/Named-entity_recognition#cite_note-1)[[2]](https://en.wikipedia.org/wiki/Named-entity_recognition#cite_note-2)



**Contents**

* [1Named-entity recognition platforms](https://en.wikipedia.org/wiki/Named-entity_recognition#Named-entity_recognition_platforms)
* [2Problem definition](https://en.wikipedia.org/wiki/Named-entity_recognition#Problem_definition)
  + [2.1Formal evaluation](https://en.wikipedia.org/wiki/Named-entity_recognition#Formal_evaluation)
* [3Approaches](https://en.wikipedia.org/wiki/Named-entity_recognition#Approaches)
* [4Problem domains](https://en.wikipedia.org/wiki/Named-entity_recognition#Problem_domains)
* [5Current challenges and research](https://en.wikipedia.org/wiki/Named-entity_recognition#Current_challenges_and_research)
* [6See also](https://en.wikipedia.org/wiki/Named-entity_recognition#See_also)
* [7References](https://en.wikipedia.org/wiki/Named-entity_recognition#References)

Named-entity recognition platforms[[edit](https://en.wikipedia.org/w/index.php?title=Named-entity_recognition&action=edit&section=1)]

Notable NER platforms include:

* [GATE](https://en.wikipedia.org/wiki/General_Architecture_for_Text_Engineering) supports NER across many languages and domains out of the box, usable via a [graphical interface](https://en.wikipedia.org/wiki/GUI) and a [Java](https://en.wikipedia.org/wiki/Java_(programming_language)) API.
* [OpenNLP](https://en.wikipedia.org/wiki/OpenNLP) includes rule-based and statistical named-entity recognition.
* [SpaCy](https://en.wikipedia.org/wiki/SpaCy) features fast statistical NER as well as an open-source named-entity visualizer.

Problem definition[[edit](https://en.wikipedia.org/w/index.php?title=Named-entity_recognition&action=edit&section=2)]

In the expression [*named entity*](https://en.wikipedia.org/wiki/Named_entity), the word *named* restricts the task to those entities for which one or many strings, such as words or phrases, stands (fairly) consistently for some referent. This is closely related to [rigid designators](https://en.wikipedia.org/wiki/Rigid_designator), as defined by [Kripke](https://en.wikipedia.org/wiki/Saul_Kripke" \o "Saul Kripke),[[3]](https://en.wikipedia.org/wiki/Named-entity_recognition#cite_note-3)[[4]](https://en.wikipedia.org/wiki/Named-entity_recognition#cite_note-4) although in practice NER deals with many names and referents that are not philosophically "rigid". For instance, the *automotive company created by Henry Ford in 1903* can be referred to as *Ford* or *Ford Motor Company*, although "Ford" can refer to many other entities as well (see [Ford](https://en.wikipedia.org/wiki/Ford_(disambiguation))). Rigid designators include proper names as well as terms for certain biological species and substances,[[5]](https://en.wikipedia.org/wiki/Named-entity_recognition#cite_note-5) but exclude pronouns (such as "it"; see [coreference resolution](https://en.wikipedia.org/wiki/Coreference_resolution)), descriptions that pick out a referent by its properties (see also [De dicto and de re](https://en.wikipedia.org/wiki/De_dicto_and_de_re)), and names for kinds of things as opposed to individuals (for example "Bank").

Full named-entity recognition is often broken down, conceptually and possibly also in implementations,[[6]](https://en.wikipedia.org/wiki/Named-entity_recognition#cite_note-6) as two distinct problems: detection of names, and [classification](https://en.wikipedia.org/wiki/Statistical_classification) of the names by the type of entity they refer to (e.g. person, organization, location and other[[7]](https://en.wikipedia.org/wiki/Named-entity_recognition#cite_note-conll03intro-7)). The first phase is typically simplified to a segmentation problem: names are defined to be contiguous spans of tokens, with no nesting, so that "Bank of America" is a single name, disregarding the fact that inside this name, the substring "America" is itself a name. This segmentation problem is formally similar to [chunking](https://en.wikipedia.org/wiki/Shallow_parsing). The second phase requires choosing an [ontology](https://en.wikipedia.org/wiki/Ontology) by which to organize categories of things.

[Temporal expressions](https://en.wikipedia.org/wiki/Temporal_annotation) and some numerical expressions (i.e., money, percentages, etc.) may also be considered as named entities in the context of the NER task. While some instances of these types are good examples of rigid designators (e.g., the year 2001) there are also many invalid ones (e.g., I take my vacations in “June”). In the first case, the year *2001* refers to the *2001st year of the Gregorian calendar*. In the second case, the month *June* may refer to the month of an undefined year (*past June*, *next June*, *every June*, etc.). It is arguable that the definition of *named entity* is loosened in such cases for practical reasons. The definition of the term *named entity* is therefore not strict and often has to be explained in the context in which it is used.[[8]](https://en.wikipedia.org/wiki/Named-entity_recognition#cite_note-8)

Certain [hierarchies](https://en.wikipedia.org/wiki/Hierarchy) of named entity types have been proposed in the literature. [BBN](https://en.wikipedia.org/wiki/BBN_Technologies) categories, proposed in 2002, is used for [*question answering*](https://en.wikipedia.org/wiki/Question_answering) and consists of 29 types and 64 subtypes.[[9]](https://en.wikipedia.org/wiki/Named-entity_recognition#cite_note-9) Sekine's extended hierarchy, proposed in 2002, is made of 200 subtypes.[[10]](https://en.wikipedia.org/wiki/Named-entity_recognition#cite_note-nlp.cs.nyu.edu-10) More recently, in 2011 Ritter used a hierarchy based on common [Freebase](https://en.wikipedia.org/wiki/Freebase_(database)) entity types in ground-breaking experiments on NER over [social media](https://en.wikipedia.org/wiki/Social_media) text.[[11]](https://en.wikipedia.org/wiki/Named-entity_recognition#cite_note-11)

**Formal evaluation**[[edit](https://en.wikipedia.org/w/index.php?title=Named-entity_recognition&action=edit&section=3)]

To evaluate the quality of a NER system's output, several measures have been defined. The usual measures are called [Precision, recall](https://en.wikipedia.org/wiki/Precision_and_recall), and [F1 score](https://en.wikipedia.org/wiki/F1_score). However, several issues remain in just how to calculate those values.

These statistical measures work reasonably well for the obvious cases of finding or missing a real entity exactly; and for finding a non-entity. However, NER can fail in many other ways, many of which are arguably "partially correct", and should not be counted as complete success or failures. For example, identifying a real entity, but:

* with fewer tokens than desired (for example, missing the last token of "John Smith, M.D.")
* with more tokens than desired (for example, including the first word of "The University of MD")
* partitioning adjacent entities differently (for example, treating "Smith, Jones Robinson" as 2 vs. 3 entities)
* assigning it a completely wrong type (for example, calling a personal name an organization)
* assigning it a related but inexact type (for example, "substance" vs. "drug", or "school" vs. "organization")
* correctly identifying an entity, when what the user wanted was a smaller- or larger-scope entity (for example, identifying "James Madison" as an personal name, when it's part of "James Madison University". Some NER systems impose the restriction that entities may never overlap or nest, which means that in some cases one must make arbitrary or task-specific choices.

One overly simple method of measuring accuracy, is merely to count what fraction of all tokens in the text were correctly or incorrectly identified as part of entity references (or as being entities of the correct type). This suffers from at least two problems: First, the vast majority of tokens in real-world text are not part of entity names, so the baseline accuracy (always predict "not an entity") is extravagantly high, typically >90%; and second, mispredicting the full span of an entity name is not properly penalized (finding only a person's first name when their last name follows might be scored as ½ accuracy).

In academic conferences such as CoNLL, a variant of the [F1 score](https://en.wikipedia.org/wiki/F1_score) has been defined as follows:[[7]](https://en.wikipedia.org/wiki/Named-entity_recognition#cite_note-conll03intro-7)

* [Precision](https://en.wikipedia.org/wiki/Precision_and_recall) is the number of predicted entity name spans that line up *exactly* with spans in the [gold standard](https://en.wikipedia.org/wiki/Ground_truth#Statistics_and_machine_learning) evaluation data. I.e. when [Person Hans] [Person Blick] is predicted but [Person Hans Blick] was required, precision for the predicted name is zero. Precision is then averaged over all predicted entity names.
* Recall is similarly the number of names in the gold standard that appear at exactly the same location in the predictions.
* F1 score is the [harmonic mean](https://en.wikipedia.org/wiki/Harmonic_mean) of these two.

It follows from the above definition that any prediction that misses a single token, includes a spurious token, or has the wrong class, is a hard error and does not contribute positively to either precision or recall. Thus, this measure may be said to be pessimistic: it can be the case that many "errors" are close to correct, and might be adequate for a given purpose. For example, one system might always omit titles such as "Ms." or "Ph.D.", but be compared to a system or ground-truth data that expects titles to be included. In that case, every such name is treated as an error. Because of such issues, it is important actually to examine the kinds of errors, and decide how important they are given one's goals and requirements.

Evaluation models based on a token-by-token matching have been proposed.[[12]](https://en.wikipedia.org/wiki/Named-entity_recognition#cite_note-12) Such models may given partial credit for overlapping matches (such as using the [Intersection over Union](https://en.wikipedia.org/wiki/Jaccard_index) criterion. They allow a finer grained evaluation and comparison of extraction systems.

Approaches[[edit](https://en.wikipedia.org/w/index.php?title=Named-entity_recognition&action=edit&section=4)]

NER systems have been created that use linguistic [grammar](https://en.wikipedia.org/wiki/Formal_grammar)-based techniques as well as [statistical models](https://en.wikipedia.org/wiki/Statistical_model) such as [machine learning](https://en.wikipedia.org/wiki/Machine_learning). Hand-crafted grammar-based systems typically obtain better precision, but at the cost of lower recall and months of work by experienced [computational linguists](https://en.wikipedia.org/wiki/Computational_linguistics).[[13]](https://en.wikipedia.org/wiki/Named-entity_recognition#cite_note-13) Statistical NER systems typically require a large amount of manually [annotated](https://en.wikipedia.org/wiki/Annotation) training data. [Semisupervised](https://en.wikipedia.org/wiki/Semisupervised_learning" \o "Semisupervised learning) approaches have been suggested to avoid part of the annotation effort.[[14]](https://en.wikipedia.org/wiki/Named-entity_recognition#cite_note-phraseclust-14)[[15]](https://en.wikipedia.org/wiki/Named-entity_recognition#cite_note-15)

Many different classifier types have been used to perform machine-learned NER, with [conditional random fields](https://en.wikipedia.org/wiki/Conditional_random_field) being a typical choice.[[16]](https://en.wikipedia.org/wiki/Named-entity_recognition#cite_note-16)

Problem domains[[edit](https://en.wikipedia.org/w/index.php?title=Named-entity_recognition&action=edit&section=5)]

In 2001, research indicated that even state-of-the-art NER systems were brittle, meaning that NER systems developed for one domain did not typically perform well on other domains.[[17]](https://en.wikipedia.org/wiki/Named-entity_recognition#cite_note-17) Considerable effort is involved in tuning NER systems to perform well in a new domain; this is true for both rule-based and trainable statistical systems.

Early work in NER systems in the 1990s was aimed primarily at extraction from journalistic articles. Attention then turned to processing of military dispatches and reports. Later stages of the [automatic content extraction](https://en.wikipedia.org/wiki/Automatic_content_extraction) (ACE) evaluation also included several types of informal text styles, such as [weblogs](https://en.wikipedia.org/wiki/Weblog) and [text transcripts](https://en.wikipedia.org/wiki/Transcription_(linguistics)) from conversational telephone speech conversations. Since about 1998, there has been a great deal of interest in entity identification in the [molecular biology](https://en.wikipedia.org/wiki/Molecular_biology), [bioinformatics](https://en.wikipedia.org/wiki/Bioinformatics), and medical [natural language processing](https://en.wikipedia.org/wiki/Natural_language_processing) communities. The most common entity of interest in that domain has been names of [genes](https://en.wikipedia.org/wiki/Gene) and gene products. There has been also considerable interest in the recognition of [chemical entities](https://en.wikipedia.org/wiki/Chemical_Entities_of_Biological_Interest) and drugs in the context of the CHEMDNER competition, with 27 teams participating in this task.[[18]](https://en.wikipedia.org/wiki/Named-entity_recognition#cite_note-18)

Current challenges and research[[edit](https://en.wikipedia.org/w/index.php?title=Named-entity_recognition&action=edit&section=6)]

Despite the high F1 numbers reported on the MUC-7 dataset, the problem of named-entity recognition is far from being solved. The main efforts are directed to reducing the annotation labor by employing [semi-supervised learning](https://en.wikipedia.org/wiki/Semi-supervised_learning),[[14]](https://en.wikipedia.org/wiki/Named-entity_recognition#cite_note-phraseclust-14)[[19]](https://en.wikipedia.org/wiki/Named-entity_recognition#cite_note-19) robust performance across domains[[20]](https://en.wikipedia.org/wiki/Named-entity_recognition#cite_note-20)[[21]](https://en.wikipedia.org/wiki/Named-entity_recognition#cite_note-21) and scaling up to fine-grained entity types.[[10]](https://en.wikipedia.org/wiki/Named-entity_recognition#cite_note-nlp.cs.nyu.edu-10)[[22]](https://en.wikipedia.org/wiki/Named-entity_recognition#cite_note-22) In recent years, many projects have turned to [crowdsourcing](https://en.wikipedia.org/wiki/Crowdsourcing), which is a promising solution to obtain high-quality aggregate human judgments for supervised and semi-supervised machine learning approaches to NER.[[23]](https://en.wikipedia.org/wiki/Named-entity_recognition#cite_note-23) Another challenging task is devising models to deal with linguistically complex contexts such as Twitter and search queries.[[24]](https://en.wikipedia.org/wiki/Named-entity_recognition#cite_note-24)

There are some researchers who did some comparisons about the NER performances from different statistical models such as HMM ([hidden Markov model](https://en.wikipedia.org/wiki/Hidden_Markov_model)), ME ([maximum entropy](https://en.wikipedia.org/wiki/Principle_of_maximum_entropy)), and CRF ([conditional random fields](https://en.wikipedia.org/wiki/Conditional_random_field)), and feature sets.[[25]](https://en.wikipedia.org/wiki/Named-entity_recognition#cite_note-25) And some researchers recently proposed graph-based semi-supervised learning model for language specific NER tasks.[[26]](https://en.wikipedia.org/wiki/Named-entity_recognition#cite_note-26)

A recently emerging task of identifying "important expressions" in text and [cross-linking them to Wikipedia](https://en.wikipedia.org/wiki/Entity_linking)[[27]](https://en.wikipedia.org/wiki/Named-entity_recognition#cite_note-27)[[28]](https://en.wikipedia.org/wiki/Named-entity_recognition#cite_note-28)[[29]](https://en.wikipedia.org/wiki/Named-entity_recognition#cite_note-29) can be seen as an instance of extremely fine-grained named-entity recognition, where the types are the actual Wikipedia pages describing the (potentially ambiguous) concepts. Below is an example output of a Wikification system:

**<ENTITY** url="http://en.wikipedia.org/wiki/Michael\_I.\_Jordan"**>** Michael Jordan **</ENTITY>** is a professor at **<ENTITY** url="http://en.wikipedia.org/wiki/University\_of\_California,\_Berkeley"**>** Berkeley **</ENTITY>**

Another field that has seen progress but remains challenging is the application of NER to [Twitter](https://en.wikipedia.org/wiki/Twitter) and other microblogs.[[30]](https://en.wikipedia.org/wiki/Named-entity_recognition#cite_note-30)[[*vague*](https://en.wikipedia.org/wiki/Wikipedia:Vagueness)]

CLOUD COMPUTING QUESTIONS

1] What is cloud computing?

[[](https://meta.wikimedia.org/wiki/Special:MyLanguage/Wikipedia_Pages_Wanting_Photos)](https://meta.wikimedia.org/wiki/Special:MyLanguage/Wikipedia_Pages_Wanting_Photos)

**[Join the WPWP Campaign to help improve Wikipedia articles with photos and win a prize](https://meta.wikimedia.org/wiki/Special:MyLanguage/Wikipedia_Pages_Wanting_Photos)**

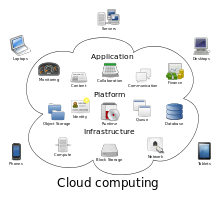
[Hide](https://en.wikipedia.org/wiki/Cloud_computing)

Cloud computing

From Wikipedia, the free encyclopedia

[Jump to navigation](https://en.wikipedia.org/wiki/Cloud_computing#mw-head)[Jump to search](https://en.wikipedia.org/wiki/Cloud_computing#searchInput)

*For the winner of the 2017 Preakness Stakes, see*[*Cloud Computing (horse)*](https://en.wikipedia.org/wiki/Cloud_Computing_(horse))*.*

[](https://en.wikipedia.org/wiki/File:Cloud_computing.svg)

Cloud computing metaphor: the group of networked elements providing services need not be individually addressed or managed by users; instead, the entire provider-managed suite of hardware and software can be thought of as an amorphous cloud.

**Cloud computing** is the on-demand availability of [computer](https://en.wikipedia.org/wiki/Computer) [system resources](https://en.wikipedia.org/wiki/System_resource), especially data storage ([cloud storage](https://en.wikipedia.org/wiki/Cloud_storage)) and [computing power](https://en.wikipedia.org/wiki/Computing_power), without direct active management by the user. The term is generally used to describe [data centers](https://en.wikipedia.org/wiki/Data_center) available to many users over the [Internet](https://en.wikipedia.org/wiki/Internet). Large clouds, predominant today, often have functions distributed over multiple locations from central [servers](https://en.wikipedia.org/wiki/Server_(computing)). If the connection to the user is relatively close, it may be designated an [edge server](https://en.wikipedia.org/wiki/Edge_server).

Clouds may be limited to a single [organization](https://en.wikipedia.org/wiki/Organization) (enterprise clouds[[1]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-1)[[2]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-aws.amazon-2)), or be available to many organizations (public cloud).

Cloud computing relies on sharing of resources to achieve coherence and [economies of scale](https://en.wikipedia.org/wiki/Economies_of_scale).

Advocates of public and hybrid clouds note that cloud computing allows companies to avoid or minimize up-front [IT infrastructure](https://en.wikipedia.org/wiki/IT_infrastructure) costs. Proponents also claim that cloud computing allows [enterprises](https://en.wikipedia.org/wiki/Company) to get their applications up and running faster, with improved manageability and less maintenance, and that it enables IT teams to more rapidly adjust resources to meet fluctuating and unpredictable demand,[[2]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-aws.amazon-2)[[3]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-3)[[4]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-4) providing the **burst computing** capability: high computing power at certain periods of peak demand.[[5]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-5)

Cloud providers typically use a "pay-as-you-go" model, which can lead to unexpected [operating expenses](https://en.wikipedia.org/wiki/Operating_expense) if [administrators](https://en.wikipedia.org/wiki/Network_administrator) are not familiarized with cloud-pricing models.[[6]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-6)

The availability of high-capacity networks, low-cost computers and storage devices as well as the widespread adoption of [hardware virtualization](https://en.wikipedia.org/wiki/Hardware_virtualization), [service-oriented architecture](https://en.wikipedia.org/wiki/Service-oriented_architecture) and [autonomic](https://en.wikipedia.org/wiki/Autonomic_computing) and [utility computing](https://en.wikipedia.org/wiki/Utility_computing) has led to growth in cloud computing.[[7]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-7)[[8]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-gartner-8)[[9]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-really-9) By 2019, [Linux](https://en.wikipedia.org/wiki/Linux_distribution) was the most widely used operating system, including in [Microsoft](https://en.wikipedia.org/wiki/Microsoft)'s offerings and is thus described as dominant.[[10]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-Linux_on_Azure-10) The Cloud Service Provider (CSP) will screen, keep up and gather data about the firewalls, intrusion identification or/and counteractive action frameworks and information stream inside the network.[[11]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-11)



**Contents**

* [1History](https://en.wikipedia.org/wiki/Cloud_computing#History)
  + [1.1Early history](https://en.wikipedia.org/wiki/Cloud_computing#Early_history)
  + [1.22000s](https://en.wikipedia.org/wiki/Cloud_computing#2000s)
  + [1.32010s](https://en.wikipedia.org/wiki/Cloud_computing#2010s)
* [2Similar concepts](https://en.wikipedia.org/wiki/Cloud_computing#Similar_concepts)
* [3Characteristics](https://en.wikipedia.org/wiki/Cloud_computing#Characteristics)
* [4Service models](https://en.wikipedia.org/wiki/Cloud_computing#Service_models)
  + [4.1Infrastructure as a service (IaaS)](https://en.wikipedia.org/wiki/Cloud_computing#Infrastructure_as_a_service_(IaaS))
  + [4.2Platform as a service (PaaS)](https://en.wikipedia.org/wiki/Cloud_computing#Platform_as_a_service_(PaaS))
  + [4.3Software as a service (SaaS)](https://en.wikipedia.org/wiki/Cloud_computing#Software_as_a_service_(SaaS))
  + [4.4Mobile "backend" as a service (MBaaS)](https://en.wikipedia.org/wiki/Cloud_computing#Mobile_%22backend%22_as_a_service_(MBaaS))
  + [4.5Serverless computing](https://en.wikipedia.org/wiki/Cloud_computing#Serverless_computing)
  + [4.6Function as a service (FaaS)](https://en.wikipedia.org/wiki/Cloud_computing#Function_as_a_service_(FaaS))
* [5Deployment models](https://en.wikipedia.org/wiki/Cloud_computing#Deployment_models)
  + [5.1Private cloud](https://en.wikipedia.org/wiki/Cloud_computing#Private_cloud)
  + [5.2Public cloud](https://en.wikipedia.org/wiki/Cloud_computing#Public_cloud)
  + [5.3Hybrid cloud](https://en.wikipedia.org/wiki/Cloud_computing#Hybrid_cloud)
  + [5.4Others](https://en.wikipedia.org/wiki/Cloud_computing#Others)
    - [5.4.1Community cloud](https://en.wikipedia.org/wiki/Cloud_computing#Community_cloud)
    - [5.4.2Distributed cloud](https://en.wikipedia.org/wiki/Cloud_computing#Distributed_cloud)
    - [5.4.3Multicloud](https://en.wikipedia.org/wiki/Cloud_computing#Multicloud)
    - [5.4.4Poly cloud](https://en.wikipedia.org/wiki/Cloud_computing#Poly_cloud)
    - [5.4.5Big Data cloud](https://en.wikipedia.org/wiki/Cloud_computing#Big_Data_cloud)
    - [5.4.6HPC cloud](https://en.wikipedia.org/wiki/Cloud_computing#HPC_cloud)
* [6Architecture](https://en.wikipedia.org/wiki/Cloud_computing#Architecture)
  + [6.1Cloud engineering](https://en.wikipedia.org/wiki/Cloud_computing#Cloud_engineering)
* [7Security and privacy](https://en.wikipedia.org/wiki/Cloud_computing#Security_and_privacy)
* [8Limitations and disadvantages](https://en.wikipedia.org/wiki/Cloud_computing#Limitations_and_disadvantages)
* [9Emerging trends](https://en.wikipedia.org/wiki/Cloud_computing#Emerging_trends)
* [10Digital forensics in the cloud](https://en.wikipedia.org/wiki/Cloud_computing#Digital_forensics_in_the_cloud)
* [11See also](https://en.wikipedia.org/wiki/Cloud_computing#See_also)
* [12References](https://en.wikipedia.org/wiki/Cloud_computing#References)
* [13Further reading](https://en.wikipedia.org/wiki/Cloud_computing#Further_reading)
* [14External links](https://en.wikipedia.org/wiki/Cloud_computing#External_links)

History[[edit](https://en.wikipedia.org/w/index.php?title=Cloud_computing&action=edit&section=1)]

|  |  |
| --- | --- |
|  | This section **may be**[**confusing or unclear**](https://en.wikipedia.org/wiki/Wikipedia:Vagueness)**to readers**. Please help us [clarify the section](https://en.wikipedia.org/wiki/Wikipedia:Please_clarify). There might be a discussion about this on [the talk page](https://en.wikipedia.org/wiki/Talk:Cloud_computing). *(July 2020) (*[*Learn how and when to remove this template message*](https://en.wikipedia.org/wiki/Help:Maintenance_template_removal)*)* |

Cloud computing was popularized with [Amazon.com](https://en.wikipedia.org/wiki/Amazon_(company)) releasing its [Elastic Compute Cloud](https://en.wikipedia.org/wiki/Amazon_Elastic_Compute_Cloud) product in 2006.[[12]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-Amazon.com-12)

References to the phrase "cloud computing" appeared as early as 1996, with the first known mention in a [Compaq](https://en.wikipedia.org/wiki/Compaq) internal document.[[13]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-13)

The cloud symbol was used to represent networks of computing equipment in the original [ARPANET](https://en.wikipedia.org/wiki/ARPANET) by as early as 1977,[[14]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-14) and the [CSNET](https://en.wikipedia.org/wiki/CSNET) by 1981[[15]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-15)—both predecessors to the Internet itself. The word *cloud* was used as a metaphor for the Internet and a standardized cloud-like shape was used to denote a network on [telephony](https://en.wikipedia.org/wiki/Telephony) schematics. With this simplification, the implication is that the specifics of how the endpoints of a network are connected are not relevant to understanding the diagram.[[16]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-16)

The term *cloud* was used to refer to platforms for [distributed computing](https://en.wikipedia.org/wiki/Distributed_computing) as early as 1993, when [Apple](https://en.wikipedia.org/wiki/Apple_Inc.) spin-off [General Magic](https://en.wikipedia.org/wiki/General_Magic) and [AT&T](https://en.wikipedia.org/wiki/AT%26T) used it in describing their (paired) [Telescript](https://en.wikipedia.org/wiki/Telescript_(programming_language)" \o "Telescript (programming language)) and PersonaLink technologies.[[17]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-17) In [*Wired's*](https://en.wikipedia.org/wiki/Wired_(magazine)) April 1994 feature "Bill and Andy's Excellent Adventure II", [Andy Hertzfeld](https://en.wikipedia.org/wiki/Andy_Hertzfeld) commented on Telescript, General Magic's distributed programming language:

"The beauty of Telescript ... is that now, instead of just having a device to program, we now have the entire Cloud out there, where a single program can go and travel to many different sources of information and create a sort of a virtual service. No one had conceived that before. The example Jim White [the designer of Telescript, [X.400](https://en.wikipedia.org/wiki/X.400) and [ASN.1](https://en.wikipedia.org/wiki/ASN.1)] uses now is a date-arranging service where a software agent goes to the flower store and orders flowers and then goes to the ticket shop and gets the tickets for the show, and everything is communicated to both parties."[[18]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-18)

**Early history**[[edit](https://en.wikipedia.org/w/index.php?title=Cloud_computing&action=edit&section=2)]

During the 1960s, the initial concepts of time-sharing became popularized via RJE ([Remote Job Entry](https://en.wikipedia.org/wiki/Remote_Job_Entry));[[19]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-19) this terminology was mostly associated with large vendors such as [IBM](https://en.wikipedia.org/wiki/IBM) and [DEC](https://en.wikipedia.org/wiki/Digital_Equipment_Corporation). Full-time-sharing solutions were available by the early 1970s on such platforms as Multics (on GE hardware), Cambridge CTSS, and the earliest UNIX ports (on DEC hardware). Yet, the "data center" model where users submitted jobs to operators to run on IBM's mainframes was overwhelmingly predominant.

In the 1990s, telecommunications companies, who previously offered primarily dedicated point-to-point data circuits, began offering [virtual private network](https://en.wikipedia.org/wiki/Virtual_private_network) (VPN) services with comparable quality of service, but at a lower cost. By switching traffic as they saw fit to balance server use, they could use overall network bandwidth more effectively.[[20]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-20) They began to use the cloud symbol to denote the demarcation point between what the provider was responsible for and what users were responsible for. Cloud computing extended this boundary to cover all servers as well as the network infrastructure.[[21]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-21) As computers became more diffused, scientists and technologists explored ways to make large-scale computing power available to more users through time-sharing.[[22]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-22) They experimented with algorithms to optimize the infrastructure, platform, and applications to prioritize CPUs and increase efficiency for end users.[[23]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-MITCorbato-23)[[24]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-24)

The use of the cloud metaphor for virtualized services dates at least to [General Magic](https://en.wikipedia.org/wiki/General_Magic) in 1994, where it was used to describe the universe of "places" that [mobile agents](https://en.wikipedia.org/wiki/Mobile_agent) in the [Telescript](https://en.wikipedia.org/wiki/Telescript_(programming_language)" \o "Telescript (programming language)) environment could go. As described by [Andy Hertzfeld](https://en.wikipedia.org/wiki/Andy_Hertzfeld):

"The beauty of [Telescript](https://en.wikipedia.org/wiki/Telescript_(programming_language)" \o "Telescript (programming language))," says [Andy](https://en.wikipedia.org/wiki/Andy_Hertzfeld), "is that now, instead of just having a device to program, we now have the entire Cloud out there, where a single program can go and travel to many different sources of information and create a sort of a virtual service."[[25]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-25)

The use of the cloud metaphor is credited to General Magic communications employee [David Hoffman](http://www.whoisdavidhoffman.com/), based on long-standing use in networking and telecom. In addition, to use by General Magic itself, it was also used in promoting [AT&T](https://en.wikipedia.org/wiki/AT%26T)'s associated PersonaLink Services.[[26]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-26)

**2000s**[[edit](https://en.wikipedia.org/w/index.php?title=Cloud_computing&action=edit&section=3)]

In August 2006, [Amazon](https://en.wikipedia.org/wiki/Amazon.com) created subsidiary [Amazon Web Services](https://en.wikipedia.org/wiki/Amazon_Web_Services) and introduced its [Elastic Compute Cloud](https://en.wikipedia.org/wiki/Amazon_Elastic_Compute_Cloud) (EC2).[[12]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-Amazon.com-12)

In April 2008, [Google](https://en.wikipedia.org/wiki/Google) released the beta version of [Google App Engine](https://en.wikipedia.org/wiki/Google_App_Engine).[[27]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-27)

In early 2008, [NASA](https://en.wikipedia.org/wiki/NASA)'s [OpenNebula](https://en.wikipedia.org/wiki/OpenNebula" \o "OpenNebula), enhanced in the RESERVOIR European Commission-funded project, became the first open-source software for deploying private and hybrid clouds, and for the federation of clouds.[[28]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-28)

By mid-2008, Gartner saw an opportunity for cloud computing "to shape the relationship among consumers of IT services, those who use IT services and those who sell them"[[29]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-29) and observed that "organizations are switching from company-owned hardware and software assets to per-use service-based models" so that the "projected shift to computing ... will result in dramatic growth in IT products in some areas and significant reductions in other areas."[[30]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-30)

In 2008, the U.S. [National Science Foundation](https://en.wikipedia.org/wiki/National_Science_Foundation) began the [Cluster Exploratory](https://en.wikipedia.org/wiki/Cluster_Exploratory) program to fund academic research using [Google](https://en.wikipedia.org/wiki/Google)-[IBM](https://en.wikipedia.org/wiki/IBM) cluster technology to analyze massive amounts of data,[[31]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-31)

**2010s**[[edit](https://en.wikipedia.org/w/index.php?title=Cloud_computing&action=edit&section=4)]

In February 2010, [Microsoft](https://en.wikipedia.org/wiki/Microsoft) released [Microsoft Azure](https://en.wikipedia.org/wiki/Microsoft_Azure), which was announced in October 2008.[[32]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-Azure-32)

In July 2010, [Rackspace Hosting](https://en.wikipedia.org/wiki/Rackspace) and [NASA](https://en.wikipedia.org/wiki/NASA) jointly launched an open-source cloud-software initiative known as [OpenStack](https://en.wikipedia.org/wiki/OpenStack). The OpenStack project intended to help organizations offering cloud-computing services running on standard hardware. The early code came from NASA's [Nebula platform](https://en.wikipedia.org/wiki/Nebula_(computing_platform)) as well as from [Rackspace's Cloud Files](https://en.wikipedia.org/wiki/Rackspace_Cloud#Cloud_Files) platform. As an open-source offering and along with other open-source solutions such as CloudStack, Ganeti, and OpenNebula, it has attracted attention by several key communities. Several studies aim at comparing these open source offerings based on a set of criteria.[[33]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-33)[[34]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-34)[[35]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-35)[[36]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-36)[[37]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-37)[[38]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-38)[[39]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-39)

On March 1, 2011, IBM announced the [IBM SmartCloud](https://en.wikipedia.org/wiki/IBM_SmartCloud) framework to support [Smarter Planet](https://en.wikipedia.org/wiki/Smarter_Planet).[[40]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-40) Among the various components of the [Smarter Computing](https://en.wikipedia.org/wiki/Smarter_Computing) foundation, cloud computing is a critical part. On June 7, 2012, Oracle announced the [Oracle Cloud](https://en.wikipedia.org/wiki/Oracle_Cloud).[[41]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-41) This cloud offering is poised to be the first to provide users with access to an integrated set of IT solutions, including the Applications ([SaaS](https://en.wikipedia.org/wiki/SaaS)), Platform ([PaaS](https://en.wikipedia.org/wiki/PaaS)), and Infrastructure ([IaaS](https://en.wikipedia.org/wiki/IaaS)) layers.[[42]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-42)[[43]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-43)[[44]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-44)

In May 2012, [Google Compute Engine](https://en.wikipedia.org/wiki/Google_Compute_Engine) was released in preview, before being rolled out into General Availability in December 2013.[[45]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-45)

In 2019, it was revealed that Linux is most used on [Microsoft Azure](https://en.wikipedia.org/wiki/Microsoft_Azure).[[10]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-Linux_on_Azure-10)

Similar concepts[[edit](https://en.wikipedia.org/w/index.php?title=Cloud_computing&action=edit&section=5)]

The goal of cloud computing is to allow users to take benefit from all of these technologies, without the need for deep knowledge about or expertise with each one of them. The cloud aims to cut costs and helps the users focus on their core business instead of being impeded by IT obstacles.[[46]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-HAM2012-46) The main enabling technology for cloud computing is [virtualization](https://en.wikipedia.org/wiki/Virtualization). Virtualization software separates a physical computing device into one or more "virtual" devices, each of which can be easily used and managed to perform computing tasks. With [operating system–level virtualization](https://en.wikipedia.org/wiki/Operating_system%E2%80%93level_virtualization) essentially creating a scalable system of multiple independent computing devices, idle computing resources can be allocated and used more efficiently. Virtualization provides the agility required to speed up IT operations and reduces cost by increasing infrastructure [utilization](https://en.wikipedia.org/wiki/Rental_utilization). Autonomic computing automates the process through which the user can provision resources [on-demand](https://en.wikipedia.org/wiki/Code_on_demand). By minimizing user involvement, automation speeds up the process, reduces labor costs and reduces the possibility of human errors.[[46]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-HAM2012-46)

Cloud computing uses concepts from utility computing to provide [metrics](https://en.wikipedia.org/wiki/Performance_metric) for the services used. Cloud computing attempts to address QoS (quality of service) and [reliability](https://en.wikipedia.org/wiki/Reliability_(computer_networking)) problems of other [grid computing](https://en.wikipedia.org/wiki/Grid_computing) models.[[46]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-HAM2012-46)

Cloud computing shares characteristics with:

* [Client–server model](https://en.wikipedia.org/wiki/Client%E2%80%93server_model)—*Client–server computing* refers broadly to any [distributed application](https://en.wikipedia.org/wiki/Distributed_application) that distinguishes between service providers (servers) and service requestors (clients).[[47]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-47)
* [Computer bureau](https://en.wikipedia.org/wiki/Computer_bureau)—A [service bureau](https://en.wikipedia.org/wiki/Service_bureau) providing computer services, particularly from the 1960s to 1980s.
* [Grid computing](https://en.wikipedia.org/wiki/Grid_computing)—A form of distributed and parallel computing, whereby a 'super and virtual computer' is composed of a [cluster](https://en.wikipedia.org/wiki/Cluster_(computing)) of networked, [loosely coupled](https://en.wikipedia.org/wiki/Loose_coupling) computers acting in concert to perform very large tasks.
* [Fog computing](https://en.wikipedia.org/wiki/Fog_computing)—Distributed computing paradigm that provides data, compute, storage and application services closer to the client or near-user edge devices, such as network routers. Furthermore, fog computing handles data at the network level, on smart devices and on the end-user client-side (e.g. mobile devices), instead of sending data to a remote location for processing.
* [Mainframe computer](https://en.wikipedia.org/wiki/Mainframe_computer)—Powerful computers used mainly by large organizations for critical applications, typically bulk data processing such as [census](https://en.wikipedia.org/wiki/Census); industry and consumer statistics; police and secret intelligence services; [enterprise resource planning](https://en.wikipedia.org/wiki/Enterprise_resource_planning); and financial [transaction processing](https://en.wikipedia.org/wiki/Transaction_processing).
* [Utility computing](https://en.wikipedia.org/wiki/Utility_computing)—The "packaging of [computing resources](https://en.wikipedia.org/wiki/Computational_resource), such as computation and storage, as a metered service similar to a traditional public utility, such as electricity."[[48]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-It's_you've-48)[[49]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-49)
* [Peer-to-peer](https://en.wikipedia.org/wiki/Peer-to-peer)—A distributed architecture without the need for central coordination. Participants are both suppliers and consumers of resources (in contrast to the traditional client-server model).
* [Green computing](https://en.wikipedia.org/wiki/Green_computing)
* [Cloud sandbox](https://en.wikipedia.org/wiki/Sandbox_(Cloud))—A live, isolated computer environment in which a program, code or file can run without affecting the application in which it runs.

Characteristics[[edit](https://en.wikipedia.org/w/index.php?title=Cloud_computing&action=edit&section=6)]

Cloud computing exhibits the following key characteristics:

* Agility for organizations may be improved, as cloud computing may increase users' flexibility with re-provisioning, adding, or expanding technological infrastructure resources.
* Cost reductions are claimed by cloud providers. A public-cloud delivery model converts [capital expenditures](https://en.wikipedia.org/wiki/Capital_expenditure) (e.g., buying servers) to [operational expenditure](https://en.wikipedia.org/wiki/Operational_expenditure).[[50]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-50) This purportedly lowers [barriers to entry](https://en.wikipedia.org/wiki/Barriers_to_entry), as infrastructure is typically provided by a third party and need not be purchased for one-time or infrequent intensive computing tasks. Pricing on a utility computing basis is "fine-grained", with usage-based billing options. As well, less in-house IT skills are required for implementation of projects that use cloud computing.[[51]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-idc-51) The e-FISCAL project's state-of-the-art repository[[52]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-52) contains several articles looking into cost aspects in more detail, most of them concluding that costs savings depend on the type of activities supported and the type of infrastructure available in-house.
* [Device and location independence](https://en.wikipedia.org/wiki/Device_independence)[[53]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-yarmis-53) enable users to access systems using a web browser regardless of their location or what device they use (e.g., PC, mobile phone). As infrastructure is off-site (typically provided by a third-party) and accessed via the Internet, users can connect to it from anywhere.[[51]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-idc-51)
* [Maintenance](https://en.wikipedia.org/wiki/Software_maintenance) of cloud computing applications is easier, because they do not need to be installed on each user's computer and can be accessed from different places (e.g., different work locations, while travelling, etc.).
* [Multitenancy](https://en.wikipedia.org/wiki/Multitenancy) enables sharing of resources and costs across a large pool of users thus allowing for:
  + centralization of infrastructure in locations with lower costs (such as real estate, electricity, etc.)
  + peak-load capacity increases (users need not engineer and pay for the resources and equipment to meet their highest possible load-levels)
  + utilisation and efficiency improvements for systems that are often only 10–20% utilised.[[54]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-amazon-54)[[55]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-55)
* [Performance](https://en.wikipedia.org/wiki/Computer_performance) is monitored by IT experts from the service provider, and consistent and loosely coupled architectures are constructed using [web services](https://en.wikipedia.org/wiki/Web_services) as the system interface.[[51]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-idc-51)[[56]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-56)
* [Productivity](https://en.wikipedia.org/wiki/Productivity) may be increased when multiple users can work on the same data simultaneously, rather than waiting for it to be saved and emailed. Time may be saved as information does not need to be re-entered when fields are matched, nor do users need to install application software upgrades to their computer.[[57]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-Smith2013-57)
* Availability improves with the use of multiple redundant sites, which makes well-designed cloud computing suitable for [business continuity](https://en.wikipedia.org/wiki/Business_continuity) and [disaster recovery](https://en.wikipedia.org/wiki/Disaster_recovery).[[58]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-58)
* Scalability and [elasticity](https://en.wikipedia.org/wiki/Elasticity_(cloud_computing)) via dynamic ("on-demand") [provisioning](https://en.wikipedia.org/wiki/Provisioning) of resources on a fine-grained, self-service basis in near real-time[[59]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-vmstartuptime2012-59)[[60]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-60) (Note, the VM startup time varies by VM type, location, OS and cloud providers[[59]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-vmstartuptime2012-59)), without users having to engineer for peak loads.[[61]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-61)[[62]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-62)[[63]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-He_15%E2%80%9322-63) This gives the ability to scale up when the usage need increases or down if resources are not being used.[[64]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-64) Emerging approaches for managing elasticity include the use of machine learning techniques to propose efficient elasticity models.[[65]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-ElsevierElasticity-65)
* [Security](https://en.wikipedia.org/wiki/Computer_security) can improve due to centralization of data, increased security-focused resources, etc., but concerns can persist about loss of control over certain sensitive data, and the lack of security for stored [kernels](https://en.wikipedia.org/wiki/Kernel_(operating_system)). Security is often as good as or better than other traditional systems, in part because service providers are able to devote resources to solving security issues that many customers cannot afford to tackle or which they lack the technical skills to address.[[66]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-66) However, the complexity of security is greatly increased when data is distributed over a wider area or over a greater number of devices, as well as in multi-tenant systems shared by unrelated users. In addition, user access to security [audit logs](https://en.wikipedia.org/wiki/Audit_log) may be difficult or impossible. Private cloud installations are in part motivated by users' desire to retain control over the infrastructure and avoid losing control of information security.

The [National Institute of Standards and Technology](https://en.wikipedia.org/wiki/National_Institute_of_Standards_and_Technology)'s definition of cloud computing identifies "five essential characteristics":

*On-demand self-service.* A consumer can unilaterally provision computing capabilities, such as server time and network storage, as needed automatically without requiring human interaction with each service provider.

*Broad network access.* Capabilities are available over the network and accessed through standard mechanisms that promote use by heterogeneous thin or thick client platforms (e.g., mobile phones, tablets, [laptops](https://en.wikipedia.org/wiki/Laptop), and workstations).

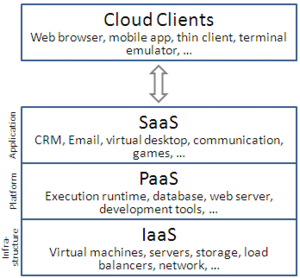
[*Resource pooling*](https://en.wikipedia.org/wiki/Pooling_(resource_management))*.* The provider's computing resources are pooled to serve multiple consumers using a multi-tenant model, with different physical and virtual resources dynamically assigned and reassigned according to consumer demand.

*Rapid elasticity.* Capabilities can be elastically provisioned and released, in some cases automatically, to scale rapidly outward and inward commensurate with demand. To the consumer, the capabilities available for provisioning often appear unlimited and can be appropriated in any quantity at any time.

*Measured service.* Cloud systems automatically control and optimize resource use by leveraging a metering capability at some level of abstraction appropriate to the type of service (e.g., storage, processing, bandwidth, and active user accounts). Resource usage can be monitored, controlled, and reported, providing transparency for both the provider and consumer of the utilized service.

— *National Institute of Standards and Technology*[[67]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-nist-67)

Service models[[edit](https://en.wikipedia.org/w/index.php?title=Cloud_computing&action=edit&section=7)]

[](https://en.wikipedia.org/wiki/File:Cloud_computing_layers.png)

Cloud computing service models arranged as layers in a stack

Though [service-oriented architecture](https://en.wikipedia.org/wiki/Service-oriented_architecture) advocates "Everything as a service" (with the acronyms **EaaS** or **XaaS**,[[68]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-68) or simply **[aas](https://en.wikipedia.org/wiki/As_a_service" \o "As a service)**), cloud-computing providers offer their "services" according to different models, of which the three standard models per [NIST](https://en.wikipedia.org/wiki/NIST) are Infrastructure as a Service (IaaS), Platform as a Service (PaaS), and Software as a Service (SaaS).[[67]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-nist-67) These models offer increasing abstraction; they are thus often portrayed as *layers* in a [stack](https://en.wikipedia.org/wiki/Solution_stack): infrastructure-, platform- and software-as-a-service, but these need not be related. For example, one can provide SaaS implemented on physical machines (bare metal), without using underlying PaaS or IaaS layers, and conversely one can run a program on IaaS and access it directly, without wrapping it as SaaS.

**Infrastructure as a service (IaaS)**[[edit](https://en.wikipedia.org/w/index.php?title=Cloud_computing&action=edit&section=8)]

*Main article:*[*Infrastructure as a service*](https://en.wikipedia.org/wiki/Infrastructure_as_a_service)

"Infrastructure as a service" (IaaS) refers to online services that provide high-level [APIs](https://en.wikipedia.org/wiki/Api) used to [abstract](https://en.wikipedia.org/wiki/Abstraction_(computer_science)) various low-level details of underlying network infrastructure like physical computing resources, location, data partitioning, scaling, security, backup, etc. A [hypervisor](https://en.wikipedia.org/wiki/Hypervisor) runs the virtual machines as guests. Pools of hypervisors within the cloud operational system can support large numbers of virtual machines and the ability to scale services up and down according to customers' varying requirements. Linux containers run in isolated partitions of a single [Linux kernel](https://en.wikipedia.org/wiki/Linux_kernel) running directly on the physical hardware. Linux [cgroups](https://en.wikipedia.org/wiki/Cgroups" \o "Cgroups) and namespaces are the underlying Linux kernel technologies used to isolate, secure and manage the containers. Containerisation offers higher performance than virtualization because there is no hypervisor overhead. Also, container capacity auto-scales dynamically with computing load, which eliminates the problem of over-provisioning and enables usage-based billing.[[69]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-69) IaaS clouds often offer additional resources such as a virtual-machine [disk-image](https://en.wikipedia.org/wiki/Disk_image) library, raw [block storage](https://en.wikipedia.org/wiki/Block_storage), file or [object storage](https://en.wikipedia.org/wiki/Object_storage), firewalls, load balancers, IP addresses, [virtual local area networks](https://en.wikipedia.org/wiki/VLAN) (VLANs), and software bundles.[[70]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-DHAC-70)

The [NIST](https://en.wikipedia.org/wiki/NIST)'s definition of cloud computing describes IaaS as "where the consumer is able to deploy and run arbitrary software, which can include operating systems and applications. The consumer does not manage or control the underlying cloud infrastructure but has control over operating systems, storage, and deployed applications; and possibly limited control of select networking components (e.g., host firewalls)."[[67]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-nist-67)

IaaS-cloud providers supply these resources on-demand from their large pools of equipment installed in [data centers](https://en.wikipedia.org/wiki/Data_centers). For [wide-area](https://en.wikipedia.org/wiki/Wide_area_network) connectivity, customers can use either the Internet or [carrier clouds](https://en.wikipedia.org/wiki/Carrier_cloud) (dedicated [virtual private networks](https://en.wikipedia.org/wiki/Virtual_private_network)). To deploy their applications, cloud users install operating-system images and their application software on the cloud infrastructure. In this model, the cloud user patches and maintains the operating systems and the application software. Cloud providers typically bill IaaS services on a utility computing basis: cost reflects the amount of resources allocated and consumed.[[71]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-71)

**Platform as a service (PaaS)**[[edit](https://en.wikipedia.org/w/index.php?title=Cloud_computing&action=edit&section=9)]

*Main article:*[*Platform as a service*](https://en.wikipedia.org/wiki/Platform_as_a_service)

The [NIST](https://en.wikipedia.org/wiki/NIST)'s definition of cloud computing defines Platform as a Service as:[[67]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-nist-67)

The capability provided to the consumer is to deploy onto the cloud infrastructure consumer-created or acquired applications created using programming languages, libraries, services, and tools supported by the provider. The consumer does not manage or control the underlying cloud infrastructure including network, servers, operating systems, or storage, but has control over the deployed applications and possibly configuration settings for the application-hosting environment.

PaaS vendors offer a development environment to application developers. The provider typically develops toolkit and standards for development and channels for distribution and payment. In the PaaS models, cloud providers deliver a [computing platform](https://en.wikipedia.org/wiki/Computing_platform), typically including operating system, programming-language execution environment, database, and web server. Application developers develop and run their software on a cloud platform instead of directly buying and managing the underlying hardware and software layers. With some PaaS, the underlying computer and storage resources scale automatically to match application demand so that the cloud user does not have to allocate resources manually.[[72]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-72)[[*need quotation to verify*](https://en.wikipedia.org/wiki/Wikipedia:Verifiability)]

Some integration and data management providers also use specialized applications of PaaS as delivery models for data. Examples include **iPaaS (Integration Platform as a Service)** and **dPaaS (Data Platform as a Service)**. iPaaS enables customers to develop, execute and govern integration flows.[[73]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-GartnerGlossary-73) Under the iPaaS integration model, customers drive the development and deployment of integrations without installing or managing any hardware or middleware.[[74]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-GartnerReferenceModel-74) dPaaS delivers integration—and data-management—products as a fully managed service.[[75]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-ITBusinessEdge-75) Under the dPaaS model, the PaaS provider, not the customer, manages the development and execution of programs by building data applications for the customer. dPaaS users access data through [data-visualization](https://en.wikipedia.org/wiki/Data_visualization) tools.[[76]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-EnterpriseCIOForum-76) Platform as a Service (PaaS) consumers do not manage or control the underlying cloud infrastructure including network, servers, operating systems, or storage, but have control over the deployed applications and possibly configuration settings for the application-hosting environment.

**Software as a service (SaaS)**[[edit](https://en.wikipedia.org/w/index.php?title=Cloud_computing&action=edit&section=10)]

*Main article:*[*Software as a service*](https://en.wikipedia.org/wiki/Software_as_a_service)

The [NIST](https://en.wikipedia.org/wiki/NIST)'s definition of cloud computing defines Software as a Service as:[[67]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-nist-67)

The capability provided to the consumer is to use the provider's applications running on a [cloud infrastructure](https://en.wikipedia.org/wiki/Cloud_infrastructure). The applications are accessible from various client devices through either a thin client interface, such as a web browser (e.g., web-based email), or a program interface. The consumer does not manage or control the underlying cloud infrastructure including network, servers, operating systems, storage, or even individual application capabilities, with the possible exception of limited user-specific application configuration settings.

In the software as a service (SaaS) model, users gain access to application software and [databases](https://en.wikipedia.org/wiki/Databases). Cloud providers manage the infrastructure and platforms that run the applications. SaaS is sometimes referred to as "on-demand software" and is usually priced on a pay-per-use basis or using a subscription fee.[[77]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-77) In the SaaS model, cloud providers install and operate application software in the cloud and cloud users access the software from cloud clients. Cloud users do not manage the cloud infrastructure and platform where the application runs. This eliminates the need to install and run the application on the cloud user's own computers, which simplifies maintenance and support. Cloud applications differ from other applications in their scalability—which can be achieved by cloning tasks onto multiple [virtual machines](https://en.wikipedia.org/wiki/Virtual_machines) at run-time to meet changing work demand.[[78]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-hamdaqa-78) [Load balancers](https://en.wikipedia.org/wiki/Load_balancer) distribute the work over the set of virtual machines. This process is transparent to the cloud user, who sees only a single [access-point](https://en.wikipedia.org/w/index.php?title=Access-point&action=edit&redlink=1). To accommodate a large number of cloud users, cloud applications can be [*multitenant*](https://en.wikipedia.org/wiki/Multitenant), meaning that any machine may serve more than one cloud-user organization.

The pricing model for SaaS applications is typically a monthly or yearly flat fee per user,[[79]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-Chou-79) so prices become scalable and adjustable if users are added or removed at any point. It may also be free.[[80]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-80) Proponents claim that SaaS gives a [business](https://en.wikipedia.org/wiki/Business) the potential to reduce IT operational costs by [outsourcing](https://en.wikipedia.org/wiki/Outsourcing) hardware and software maintenance and support to the cloud provider. This enables the business to reallocate IT operations costs away from hardware/software spending and from personnel expenses, towards meeting other goals. In addition, with applications hosted centrally, updates can be released without the need for users to install new software. One drawback of SaaS comes with storing the users' data on the cloud provider's server. As a result,[[*citation needed*](https://en.wikipedia.org/wiki/Wikipedia:Citation_needed)] there could be [unauthorized access](https://en.wikipedia.org/w/index.php?title=Unauthorized_access&action=edit&redlink=1) to the data.[[*citation needed*](https://en.wikipedia.org/wiki/Wikipedia:Citation_needed)] Examples of applications offered as SaaS are [games](https://en.wikipedia.org/wiki/Cloud_gaming) and productivity software like Google Docs and Word Online. SaaS applications may be integrated with [cloud storage](https://en.wikipedia.org/wiki/Cloud_storage) or [File hosting services](https://en.wikipedia.org/wiki/File_hosting_service), which is the case with Google Docs being integrated with Google Drive and Word Online being integrated with Onedrive.[[*citation needed*](https://en.wikipedia.org/wiki/Wikipedia:Citation_needed)]

**Mobile "backend" as a service (MBaaS)**[[edit](https://en.wikipedia.org/w/index.php?title=Cloud_computing&action=edit&section=11)]

*Main article:*[*Mobile backend as a service*](https://en.wikipedia.org/wiki/Mobile_backend_as_a_service)

In the mobile "backend" as a service (m) model, also known as **backend as a service (BaaS)**, [web app](https://en.wikipedia.org/wiki/Web_app) and [mobile app](https://en.wikipedia.org/wiki/Mobile_app) developers are provided with a way to link their applications to [cloud storage](https://en.wikipedia.org/wiki/Cloud_storage) and cloud computing services with [application programming interfaces](https://en.wikipedia.org/wiki/Application_programming_interface) (APIs) exposed to their applications and custom [software development kits](https://en.wikipedia.org/wiki/Software_development_kit) (SDKs). Services include user management, [push notifications](https://en.wikipedia.org/wiki/Push_technology), integration with [social networking services](https://en.wikipedia.org/wiki/Social_networking_service)[[81]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-PandoDailyAP-81) and more. This is a relatively recent model in cloud computing,[[82]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-Williams1-82) with most BaaS [startups](https://en.wikipedia.org/wiki/Startup_company) dating from 2011 or later[[83]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-Tan12-83)[[84]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-Rowinski11-84)[[85]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-Mishra-85) but trends indicate that these services are gaining significant mainstream traction with enterprise consumers.[[86]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-built.io-86)

**Serverless computing**[[edit](https://en.wikipedia.org/w/index.php?title=Cloud_computing&action=edit&section=12)]

*Main article:*[*Serverless computing*](https://en.wikipedia.org/wiki/Serverless_computing)

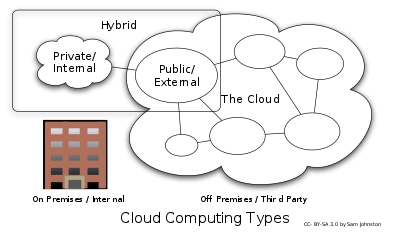
Serverless computing is a cloud computing code [execution](https://en.wikipedia.org/wiki/Execution_(computing)) model in which the cloud provider fully manages starting and stopping [virtual machines](https://en.wikipedia.org/wiki/Virtual_machines) as necessary to serve requests, and requests are billed by an abstract measure of the resources required to satisfy the request, rather than per virtual machine, per hour.[[87]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-techcrunch-lambda-87) Despite the name, it does not actually involve running code without servers.[[87]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-techcrunch-lambda-87) Serverless computing is so named because the business or person that owns the system does not have to purchase, rent or provision servers or virtual machines for the [back-end](https://en.wikipedia.org/wiki/Back-end_database) code to run on.

**Function as a service (FaaS)**[[edit](https://en.wikipedia.org/w/index.php?title=Cloud_computing&action=edit&section=13)]

*Main article:*[*Function as a service*](https://en.wikipedia.org/wiki/Function_as_a_service)

Function as a service (FaaS) is a service-hosted remote procedure call that leverages serverless computing to enable the deployment of individual functions in the cloud that run in response to events.[[88]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-88) FaaS is included under the broader term [*serverless computing*](https://en.wikipedia.org/wiki/Serverless_computing), but the terms may also be used interchangeably.[[89]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-89)

Deployment models[[edit](https://en.wikipedia.org/w/index.php?title=Cloud_computing&action=edit&section=14)]

[](https://en.wikipedia.org/wiki/File:Cloud_computing_types.svg)

Cloud computing types

**Private cloud**[[edit](https://en.wikipedia.org/w/index.php?title=Cloud_computing&action=edit&section=15)]

Private cloud is cloud infrastructure operated solely for a single organization, whether managed internally or by a third party, and hosted either internally or externally.[[67]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-nist-67) Undertaking a private cloud project requires significant engagement to virtualize the business environment, and requires the organization to reevaluate decisions about existing resources. It can improve business, but every step in the project raises security issues that must be addressed to prevent serious vulnerabilities. Self-run [data centers](https://en.wikipedia.org/wiki/Data_center)[[90]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-90) are generally capital intensive. They have a significant physical footprint, requiring allocations of space, hardware, and environmental controls. These assets have to be refreshed periodically, resulting in additional capital expenditures. They have attracted criticism because users "still have to buy, build, and manage them" and thus do not benefit from less hands-on management,[[91]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-iwpc-91) essentially "[lacking] the economic model that makes cloud computing such an intriguing concept".[[92]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-92)[[93]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-93)

**Public cloud**[[edit](https://en.wikipedia.org/w/index.php?title=Cloud_computing&action=edit&section=16)]

*For a comparison of cloud-computing software and providers, see*[*Cloud-computing comparison*](https://en.wikipedia.org/wiki/Cloud-computing_comparison)

A cloud is called a "public cloud" when the services are rendered over a network that is open for public use. Public cloud services may be free.[[94]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-94) Technically there may be little or no difference between public and private cloud architecture, however, security consideration may be substantially different for services (applications, storage, and other resources) that are made available by a service provider for a public audience and when communication is effected over a non-trusted network. Generally, public cloud service providers like [Amazon Web Services (AWS)](https://en.wikipedia.org/wiki/Amazon_Web_Services), [IBM Cloud](https://en.wikipedia.org/wiki/IBM_Cloud), [Oracle](https://en.wikipedia.org/wiki/Oracle_Cloud), [Microsoft](https://en.wikipedia.org/wiki/Microsoft_Azure), [Google](https://en.wikipedia.org/wiki/Google_Cloud_Platform), and [Alibaba](https://en.wikipedia.org/wiki/Alibaba_Cloud) own and operate the infrastructure at their [data center](https://en.wikipedia.org/wiki/Data_center) and access is generally via the Internet. AWS, Oracle, Microsoft, and Google also offer [direct connect](https://en.wikipedia.org/wiki/Direct_Connect_(protocol)) services called "AWS Direct Connect", "Oracle FastConnect", "Azure ExpressRoute", and "Cloud Interconnect" respectively, such connections require customers to purchase or lease a private connection to a peering point offered by the cloud provider.[[51]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-idc-51)[[95]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-95)

**Hybrid cloud**[[edit](https://en.wikipedia.org/w/index.php?title=Cloud_computing&action=edit&section=17)]

**Hybrid** cloud is a composition of a public cloud and a private environment, such as a private cloud or on-premises resources,[[96]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-96)[[97]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-97)[[98]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-98) that remain distinct entities but are bound together, offering the benefits of multiple deployment models. Hybrid cloud can also mean the ability to connect collocation, managed and/or dedicated services with cloud resources.[[67]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-nist-67) [Gartner](https://en.wikipedia.org/wiki/Gartner) defines a hybrid cloud service as a cloud computing service that is composed of some combination of private, public and community cloud services, from different service providers.[[99]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-99) A hybrid cloud service crosses isolation and provider boundaries so that it can't be simply put in one category of private, public, or community cloud service. It allows one to extend either the capacity or the capability of a cloud service, by aggregation, integration or customization with another cloud service.

Varied use cases for hybrid cloud composition exist. For example, an organization may store sensitive client data in house on a private cloud application, but interconnect that application to a business intelligence application provided on a public cloud as a software service.[[100]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-100) This example of hybrid cloud extends the capabilities of the enterprise to deliver a specific business service through the addition of externally available public cloud services. Hybrid cloud adoption depends on a number of factors such as data security and compliance requirements, level of control needed over data, and the applications an organization uses.[[101]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-101)

Another example of hybrid cloud is one where [IT](https://en.wikipedia.org/wiki/Information_technology) organizations use public cloud computing resources to meet temporary capacity needs that can not be met by the private cloud.[[102]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-102) This capability enables hybrid clouds to employ cloud bursting for scaling across clouds.[[67]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-nist-67) Cloud bursting is an application deployment model in which an application runs in a private cloud or data center and "bursts" to a public cloud when the demand for computing capacity increases. A primary advantage of cloud bursting and a hybrid cloud model is that an organization pays for extra compute resources only when they are needed.[[103]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-103) Cloud bursting enables data centers to create an in-house IT infrastructure that supports average workloads, and use cloud resources from public or private clouds, during spikes in processing demands.[[104]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-104) The specialized model of hybrid cloud, which is built atop heterogeneous hardware, is called "Cross-platform Hybrid Cloud". A cross-platform hybrid cloud is usually powered by different CPU architectures, for example, x86-64 and ARM, underneath. Users can transparently deploy and scale applications without knowledge of the cloud's hardware diversity.[[105]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-105) This kind of cloud emerges from the rise of ARM-based system-on-chip for server-class computing.

Hybrid cloud infrastructure essentially serves to eliminate limitations inherent to the multi-access relay characteristics of private cloud networking. The advantages include enhanced runtime flexibility and adaptive memory processing unique to virtualized interface models.[[106]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-106)

**Others**[[edit](https://en.wikipedia.org/w/index.php?title=Cloud_computing&action=edit&section=18)]

**Community cloud**[[edit](https://en.wikipedia.org/w/index.php?title=Cloud_computing&action=edit&section=19)]

[Community cloud](https://en.wikipedia.org/wiki/Community_cloud) shares infrastructure between several organizations from a specific community with common concerns (security, compliance, jurisdiction, etc.), whether managed internally or by a third-party, and either hosted internally or externally. The costs are spread over fewer users than a public cloud (but more than a private cloud), so only some of the cost savings potential of cloud computing are realized.[[67]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-nist-67)

**Distributed cloud**[[edit](https://en.wikipedia.org/w/index.php?title=Cloud_computing&action=edit&section=20)]

A cloud computing platform can be assembled from a distributed set of machines in different locations, connected to a single network or hub service. It is possible to distinguish between two types of distributed clouds: public-resource computing and volunteer cloud.

* **Public-resource computing**—This type of distributed cloud results from an expansive definition of cloud computing, because they are more akin to distributed computing than cloud computing. Nonetheless, it is considered a sub-class of cloud computing.
* **Volunteer cloud**—Volunteer cloud computing is characterized as the intersection of public-resource computing and cloud computing, where a cloud computing infrastructure is built using volunteered resources. Many challenges arise from this type of infrastructure, because of the volatility of the resources used to build it and the dynamic environment it operates in. It can also be called peer-to-peer clouds, or ad-hoc clouds. An interesting effort in such direction is Cloud@Home, it aims to implement a cloud computing infrastructure using volunteered resources providing a business-model to incentivize contributions through financial restitution.[[107]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-107)

**Multicloud**[[edit](https://en.wikipedia.org/w/index.php?title=Cloud_computing&action=edit&section=21)]

*Main article: [Multicloud](https://en.wikipedia.org/wiki/Multicloud" \o "Multicloud)*

Multicloud is the use of multiple cloud computing services in a single heterogeneous architecture to reduce reliance on single vendors, increase flexibility through choice, mitigate against disasters, etc. It differs from hybrid cloud in that it refers to multiple cloud services, rather than multiple deployment modes (public, private, legacy).[[108]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-rouse-108)[[109]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-king-109)[[110]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-110)

**Poly cloud**[[edit](https://en.wikipedia.org/w/index.php?title=Cloud_computing&action=edit&section=22)]

Poly cloud refers to the use of multiple public clouds for the purpose of leveraging specific services that each provider offers. It differs from multicloud in that it is not designed to increase flexibility or mitigate against failures but is rather used to allow an organization to achieve more that could be done with a single provider.[[111]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-111)

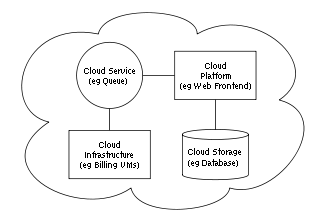
**Big Data cloud**[[edit](https://en.wikipedia.org/w/index.php?title=Cloud_computing&action=edit&section=23)]

The issues of transferring large amounts of data to the cloud as well as data security once the data is in the cloud initially hampered adoption of cloud for [big data](https://en.wikipedia.org/wiki/Big_data), but now that much data originates in the cloud and with the advent of [bare-metal servers](https://en.wikipedia.org/wiki/Bare-metal_server), the cloud has become[[112]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-112) a solution for use cases including business [analytics](https://en.wikipedia.org/wiki/Analytics) and [geospatial analysis](https://en.wikipedia.org/wiki/Geospatial_analysis).[[113]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-YANG-113)

**HPC cloud**[[edit](https://en.wikipedia.org/w/index.php?title=Cloud_computing&action=edit&section=24)]

HPC cloud refers to the use of cloud computing services and infrastructure to execute [high-performance computing](https://en.wikipedia.org/wiki/High-performance_computing) (HPC) applications.[[114]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-114) These applications consume considerable amount of computing power and memory and are traditionally executed on [clusters](https://en.wikipedia.org/wiki/Computer_cluster) of computers. In 2016 a handful of companies, including R-HPC, [Amazon Web Services](https://en.wikipedia.org/wiki/Amazon_Web_Services), [Univa](https://en.wikipedia.org/wiki/Univa" \o "Univa), [Silicon Graphics International](https://en.wikipedia.org/wiki/Silicon_Graphics_International), Sabalcore, Gomput, and [Penguin Computing](https://en.wikipedia.org/wiki/Penguin_Computing) offered a high performance computing cloud. The Penguin On Demand (POD) cloud was one of the first non-virtualized remote HPC services offered on a [pay-as-you-go](https://en.wikipedia.org/wiki/Prepayment_for_service) basis.[[115]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-115)[[116]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-116) Penguin Computing launched its HPC cloud in 2016 as alternative to Amazon's EC2 Elastic Compute Cloud, which uses virtualized computing nodes.[[117]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-117)[[118]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-118)

Architecture[[edit](https://en.wikipedia.org/w/index.php?title=Cloud_computing&action=edit&section=25)]

[](https://en.wikipedia.org/wiki/File:CloudComputingSampleArchitecture.svg)

Cloud computing sample architecture

**Cloud architecture**,[[119]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-119) the [systems architecture](https://en.wikipedia.org/wiki/Systems_architecture) of the [software systems](https://en.wikipedia.org/wiki/Software_systems) involved in the delivery of cloud computing, typically involves multiple *cloud components* communicating with each other over a loose coupling mechanism such as a messaging queue. Elastic provision implies intelligence in the use of tight or loose coupling as applied to mechanisms such as these and others.

**Cloud engineering**[[edit](https://en.wikipedia.org/w/index.php?title=Cloud_computing&action=edit&section=26)]

[**Cloud engineering**](https://en.wikipedia.org/wiki/Cloud_engineering) is the application of [engineering](https://en.wikipedia.org/wiki/Engineering) disciplines to cloud computing. It brings a systematic approach to the high-level concerns of commercialization, standardization and governance in conceiving, developing, operating and maintaining cloud computing systems. It is a multidisciplinary method encompassing contributions from diverse areas such as [systems](https://en.wikipedia.org/wiki/Systems_engineering), [software](https://en.wikipedia.org/wiki/Software_engineering), [web](https://en.wikipedia.org/wiki/Web_engineering), [performance](https://en.wikipedia.org/wiki/Performance_engineering), [information technology engineering](https://en.wikipedia.org/wiki/Information_technology_engineering), [security](https://en.wikipedia.org/wiki/Security_engineering), [platform](https://en.wikipedia.org/wiki/Platform_engineering), [risk](https://en.wikipedia.org/wiki/Risk_analysis_(engineering)), and [quality](https://en.wikipedia.org/wiki/Quality_control) engineering.

Security and privacy[[edit](https://en.wikipedia.org/w/index.php?title=Cloud_computing&action=edit&section=27)]

*Main article:*[*Cloud computing issues*](https://en.wikipedia.org/wiki/Cloud_computing_issues)

Cloud computing poses privacy concerns because the service provider can access the data that is in the cloud at any time. It could accidentally or deliberately alter or delete information.[[120]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-ryan-120) Many cloud providers can share information with third parties if necessary for purposes of law and order without a warrant. That is permitted in their privacy policies, which users must agree to before they start using cloud services. Solutions to privacy include policy and legislation as well as end-users' choices for how data is stored.[[120]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-ryan-120) Users can encrypt data that is processed or stored within the cloud to prevent unauthorized access.[[121]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-cloudid-121)[[120]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-ryan-120) [Identity management systems](https://en.wikipedia.org/wiki/Identity_management_systems) can also provide practical solutions to privacy concerns in cloud computing. These systems distinguish between authorized and unauthorized users and determine the amount of data that is accessible to each entity.[[*citation needed*](https://en.wikipedia.org/wiki/Wikipedia:Citation_needed)] The systems work by creating and describing identities, recording activities, and getting rid of unused identities.

According to the Cloud Security Alliance, the top three threats in the cloud are *Insecure Interfaces and APIs*, *Data Loss & Leakage*, and *Hardware Failure*—which accounted for 29%, 25% and 10% of all cloud security outages respectively. Together, these form shared technology vulnerabilities. In a cloud provider platform being shared by different users, there may be a possibility that information belonging to different customers resides on the same data server. Additionally, [Eugene Schultz](https://en.wikipedia.org/wiki/Eugene_Schultz), chief technology officer at Emagined Security, said that hackers are spending substantial time and effort looking for ways to penetrate the cloud. "There are some real Achilles' heels in the cloud infrastructure that are making big holes for the bad guys to get into". Because data from hundreds or thousands of companies can be stored on large cloud servers, hackers can theoretically gain control of huge stores of information through a single attack—a process he called "hyperjacking". Some examples of this include the Dropbox security breach, and iCloud 2014 leak.[[122]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-psg.hitachi-solutions.com-122) Dropbox had been breached in October 2014, having over 7 million of its users passwords stolen by hackers in an effort to get monetary value from it by Bitcoins (BTC). By having these passwords, they are able to read private data as well as have this data be indexed by search engines (making the information public).[[122]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-psg.hitachi-solutions.com-122)

There is the problem of legal ownership of the data (If a user stores some data in the cloud, can the cloud provider profit from it?). Many Terms of Service agreements are silent on the question of ownership.[[123]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-123) Physical control of the computer equipment (private cloud) is more secure than having the equipment off-site and under someone else's control (public cloud). This delivers great incentive to public cloud computing service providers to prioritize building and maintaining strong management of secure services.[[124]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-124) Some small businesses that don't have expertise in [IT](https://en.wikipedia.org/wiki/Information_technology) security could find that it's more secure for them to use a public cloud. There is the risk that end users do not understand the issues involved when signing on to a cloud service (persons sometimes don't read the many pages of the terms of service agreement, and just click "Accept" without reading). This is important now that cloud computing is becoming popular and required for some services to work, for example for an [intelligent personal assistant](https://en.wikipedia.org/wiki/Intelligent_personal_assistant) (Apple's [Siri](https://en.wikipedia.org/wiki/Siri) or [Google Now](https://en.wikipedia.org/wiki/Google_Now)). Fundamentally, private cloud is seen as more secure with higher levels of control for the owner, however public cloud is seen to be more flexible and requires less time and money investment from the user.[[125]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-125)

Limitations and disadvantages[[edit](https://en.wikipedia.org/w/index.php?title=Cloud_computing&action=edit&section=28)]

According to [Bruce Schneier](https://en.wikipedia.org/wiki/Bruce_Schneier), "The downside is that you will have limited customization options. Cloud computing is cheaper because of [economics of scale](https://en.wikipedia.org/wiki/Economies_of_scale), and—like any outsourced task—you tend to get what you want. A restaurant with a limited menu is cheaper than a personal chef who can cook anything you want. Fewer options at a much cheaper price: it's a feature, not a bug." He also suggests that "the cloud provider might not meet your legal needs" and that businesses need to weigh the benefits of cloud computing against the risks.[[126]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-126) In cloud computing, the control of the back end infrastructure is limited to the cloud vendor only. Cloud providers often decide on the management policies, which moderates what the cloud users are able to do with their deployment.[[127]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-127) Cloud users are also limited to the control and management of their applications, data and services.[[128]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-The_real_limits_of_cloud_computing-128) This includes [data caps](https://en.wikipedia.org/wiki/Data_cap), which are placed on cloud users by the cloud vendor allocating a certain amount of bandwidth for each customer and are often shared among other cloud users.[[128]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-The_real_limits_of_cloud_computing-128)

Privacy and [confidentiality](https://en.wikipedia.org/wiki/Confidentiality) are big concerns in some activities. For instance, sworn translators working under the stipulations of an [NDA](https://en.wikipedia.org/wiki/Non-disclosure_agreement), might face problems regarding [sensitive data](https://en.wikipedia.org/wiki/Sensitive_data) that are not [encrypted](https://en.wikipedia.org/wiki/Encrypt).[[129]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-129)

Cloud computing is beneficial to many enterprises; it lowers costs and allows them to focus on competence instead of on matters of IT and infrastructure. Nevertheless, cloud computing has proven to have some limitations and disadvantages, especially for smaller business operations, particularly regarding security and downtime. Technical outages are inevitable and occur sometimes when cloud service providers (CSPs) become overwhelmed in the process of serving their clients. This may result in temporary business suspension. Since this technology's systems rely on the Internet, an individual cannot access their applications, server or data from the cloud during an outage.[[130]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-130)However, many large enterprises maintain at least two internet providers, using different entry points into their workplaces, some even use 4G as a third fallback.

Emerging trends[[edit](https://en.wikipedia.org/w/index.php?title=Cloud_computing&action=edit&section=29)]

Cloud computing is still a subject of research.[[131]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-ghc-131) A driving factor in the evolution of cloud computing has been [chief technology officers](https://en.wikipedia.org/wiki/Chief_technology_officer) seeking to minimize risk of internal outages and mitigate the complexity of housing network and computing hardware in-house.[[132]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-132) Major cloud technology companies invest billions of dollars per year in cloud [Research and Development](https://en.wikipedia.org/wiki/Research_and_Development). For example, in 2011 Microsoft committed 90 percent of its $9.6 billion [R&D](https://en.wikipedia.org/wiki/Research_and_development) budget to its cloud.[[133]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-133) Research by investment bank Centaur Partners in late 2015 forecasted that SaaS revenue would grow from $13.5 billion in 2011 to $32.8 billion in 2016.[[134]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-134)

Digital forensics in the cloud[[edit](https://en.wikipedia.org/w/index.php?title=Cloud_computing&action=edit&section=30)]

The issue of carrying out investigations where the cloud storage devices cannot be physically accessed has generated a number of changes to the way that digital evidence is located and collected.[[135]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-135) New process models have been developed to formalize collection.[[136]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-136)

In some scenarios existing digital forensics tools can be employed to access cloud storage as networked drives (although this is a slow process generating a large amount of internet traffic).[[*citation needed*](https://en.wikipedia.org/wiki/Wikipedia:Citation_needed)]

An alternative approach is to deploy a tool that processes in the cloud itself.[[137]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-137)

For organizations using Office 365 with an 'E5' subscription, there is the option to use Microsoft's built-in ediscovery resources, although these do not provide all the functionality that is typically required for a forensic process.[[138]](https://en.wikipedia.org/wiki/Cloud_computing#cite_note-138)

2] Which are the different layers that define cloud architecture?

Virtualization and dynamic provisioning of resources are the principles on which cloud computing works. In terms of architecture, the [cloud hosting](https://www.cloudoye.com/) can be sliced into four different layers.

**The Physical Layer:**

This layer comprises of physical servers, network and other aspects that can be physically managed and controlled.

**The Infrastructure Layer:**

This includes storage facilities, [virtualized servers](https://www.cloudoye.com/vps-hosting), and networking. [Infrastructure as a Service or IaaS points](https://www.cloudoye.com/iaas-hosting) to delivery of services in hosted format. They include hardware, network and servers, delivered to end users. Consumers can enjoy access to scalable storage and compute power as and when needed.

**Platform Layer:**

This layer includes services such as OS and Apps. It serves as a platform for development and deployment. The Platform layer provides the right platform for development and deployment of applications vital for the cloud to run smoothly.

**Application Layer:**

The Application Layer is the one that end users interact with in a direct manner. It mainly comprises of software systems delivered as service. Examples are Gmail and Dropbox. [SaaS or Software as a Service](https://www.cloudoye.com/saas-hosting) ensures delivery of software in hosted form which can be accessed by users through the internet. Configurability and scalability are the two key features of this layer. Customers can easily customize their software system using Meta data.

## Uncover the Capabilities of Cloud Technology

If you define cloud computing as those services that are available through a web interface, you have a very basic understanding of cloud. Digging into the three layers of cloud services, however, will uncover the capabilities of cloud technologies that could help your company discover ways to be more agile in how you respond to changing customer needs or market opportunities. The following will expand your cloud computing definition to include Software as a Service (SaaS), Platform as a Service (PaaS), and Infrastructure as a Service (IaaS).

### Software as a Service (SaaS)

This is the layer of the cloud that is familiar to most people. If you go to a web page and login in order to access a tool or service, you are using SaaS. Some SaaS offerings are free, or at least free for a baseline set of services. Businesses utilize SaaS on a subscription basis for multiple functions from HR, accounting and payroll, to sales, marketing and project management. While the core functionality of the application is hosted on the provider’s servers, there are some apps that require you to download software to your PC pr device to extends capabilities such as synchronization with multiple devices.

The benefits of SaaS include easy entry and startup. Most apps allow you to add or subtract users as needed so that you are only paying for what you use. The provider takes care of all software upgrades, support and uptime. Many SaaS applications can be integrated with one another to customize their use. For example, your marketing application could integrate with your prospect database, or your procurement application can integrate with your ERP software.

### Platform as a Service (PaaS)

If you aren’t a software programmer or web developer you might not think that you will ever have a use for PaaS, but this layer of the cloud is the one that can really address your unique requirements. In a cloud computing definition, PaaS simply means that this is the place where software is developed. You might use this if you need to customize your industry-specific business application, building and testing it in the cloud before deploying to the company. You could also test out a software update in a PaaS environment when there are concerns about how the update will affect your processes, a situation that could occur if you have many customizations. If your company is considering taking your business applications to the cloud, you could use PaaS to build, test and launch.

### Infrastructure as a Service (IaaS)

The physical components of your infrastructure – hardware, software, storage and network workings – are housed remotely and accessed via the internet when you utilize IaaS. The provider handles tasks like system maintenance, software updates, backup and security. IaaS is good for companies that need to scale up or down quickly, or that experience sudden changes in capacity. Subscriptions and pay-as-you-go arrangements eliminate capital expenses of deploying hardware. A part of PaaS is actually the infrastructure that goes with the platform, so there is some overlap with these two layers of the cloud

Cloud architecture is not as simple as it first seems. Cloud is the outcome of several layers of cloud architecture intelligently placed over one another. Before we move towards the various layers, take a look at the more general picture of cloud layers below -

**Hardware Layer:** This bottom most layer of cloud architecture, the hardware layer, primarily deals with all the hardware powering clouds. The hardware includes but is not restricted to routers, servers, switches, power and cooling systems.

**Infrastructure Layer:** Also called the virtualization layer, the infrastructure layer is where all the servers are pooled together into one.

**Platform Layer:** The platform layer comprises the operating system and other requisition structures and is based over the infrastructure layer.

**Application Layer:** As the name suggests, the application layer - the topmost layer - contains applications that directly interact with the end-user.

Cloud computing architecture is made of several layers for better operational efficiency. Cloud controller or CLC is at the top and is used to manage virtualized resources like servers, network and storage. Walrus is the next layer and used as a storage controller to manage the demands of the users. Cluster Controller or CC manages the virtual networking between Virtual machines and external users. Storage Controller or SC is a block-form [storage device](https://go4hosting.in/services/cloud/storage), dynamically attached by [Virtual machines](https://go4hosting.in/virtual-machine). The next layer is NC or Node Controller. It acts as a hypervisor and controls the Virtual machines activities such as execution, management and termination of many instances.

## What are the Various Types of Cloud Computing?

### Cloud Computing is of the Following three types:

**IaaS -** Infrastructure as a service. A [cloud service](https://go4hosting.in/services/indian-cloud-hosting) is said to be IaaS when the provider is responsible for creating the entire virtualization environment for the client. The provider will setup the cloud, pool resource of the server together, turn on the lights and hand the keys of the cloud to the client. It is the bottom most service that can be sold to the client.

**PaaS -**Platform as a Service. When the provider itself configures an operating system on the infra, creating a ready platform to be used for various needs, it is called the PaaS or Platform as a Service. PaaS is mostly used by web developers for launching VMs as it gives them a ready made platform to start developing applications as per their needs.

**SaaS -** Software as a service is the topmost service layer that can be sold among various layers of cloud architecture. Of all types of cloud computing, this one involves the end-user and the underlying hardware the least. In SaaS, the client is not at all concerned with the layers underpinning the cloud and only works at the topmost layer.

### Which type of Cloud Computing is the Best?

It is difficult to weigh the various types of [cloud computing](https://go4hosting.in/knowledgebase/cloud-computing/what-is-cloud-computing) on the same scale. Each service or the layer has its own characteristic advantage. Say, the IaaS is the most suitable for organizations who covet the ultimate control of their cloud platform. PaaS is more apt for users who want an Operating System or any other software pre-installed in the cloud. Even if these users were to opt for IaaS, they would reap no added benefits because their requirements are different altogether. Similarly, software as a service is meant for less proficient users, who only need an application to perform specific functions. SaaS clients only concern themselves with the applications and not the cloud architecture. Some users might not even have an idea that their service is running with cloud computing underlying it.