UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
14/170,404	01/31/2014	Rajeev PANDEY	83748040	5505

146568 7590 10/31/2019
MICRO FOCUS LLC
500 Westover Drive
#12603
Sanford, NC 27330

EXAMINER	
MEKONEN, TESFU N	
ART UNIT	**PAPER NUMBER**
2454	
NOTIFICATION DATE	**DELIVERY MODE**
10/31/2019	ELECTRONIC

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

Notice of the Office communication was sent electronically on above-indicated "Notification Date" to the following e-mail address(es):

software.ip.mail@microfocus.com

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

Ex parte RAJEEV PANDEY and MATTHEW FARINA

Appeal 2018-006424
Application 14/170,404
Technology Center 2400

Before ROBERT E. NAPPI, DENISE M. POTHIER, and
JOHN A. EVANS, *Administrative Patent Judges*.

POTHIER, *Administrative Patent Judge*.

DECISION ON APPEAL

STATEMENT OF THE CASE

Appellant[1,2] appeals under 35 U.S.C. § 134(a) from the rejections of claims 1–12 and 16–25. Appeal Br. 6. Claims 13–15 have been canceled. *Id.* at i (Claims App'x). We have jurisdiction under 35 U.S.C. § 6(b).

[1] Throughout this opinion, we refer to the Final Action (Final Act.) mailed June 23, 2017, the Appeal Brief (Appeal Br.) filed November 2, 2017, the Examiner's Answer (Ans.) mailed April 5, 2018, and the Reply Brief (Reply Br.) filed June 5, 2018.
[2] We use the word "Appellant" to refer to "applicant" as defined in 37 C.F.R. § 1.42. Appellant identifies the real party in interest as EntIT Software LLC. Appeal Br. 1.

We AFFIRM IN PART.

As background to this proceeding, the Specification states:

> An application can exist on a shared pool of resources, referred to herein as a 'cloud' for access by a client computer. . . . The resources of the cloud can be virtual instances of resources, such as a virtual machine instance. The cloud can be structured . . . to provide [a] service based on a cloud service model, such as software as a service ("SaaS"), platform as a service ("PaaS"), and infrastructure as a service ("IaaS").

Spec. ¶ 1.

Appellant's invention "relate[s] to orchestrating a cloud based on an implementation selection." *Id.* ¶ 8. One "cloud orchestration system can comprise a solution engine, a configuration engine, and an implementation engine." *Id.*, Abstract; *see id.* ¶¶ 10–20, 28, Figs. 4 (elements 404, 422, 406), 5 (elements 522, 506). "The solution engine can receive an implementation selection of a service," "the configuration engine can obtain configuration information associated with the implementation selection," and "[t]he implementation engine can communicate with an endpoint of the service based on the implementation selection and the configuration information." Spec., Abstract; *see id.* ¶¶ 10–20, 28, Fig. 4 (elements 404, 422, 406). Services include "communication, transmissions, software, [and] storage" with a network of electronic devices. *Id.* ¶ 6.

Independent claim 1 reads as follows:

> A cloud orchestration system comprising:
> a processor; and
> a non-transitory storage medium storing instructions executable on the processor to:
> cause presentation, by a marketplace, of a plurality of solutions providing different services, each solution of

> the plurality of solutions including a respective plurality of different implementation options;
>
> receive user selection of a first solution of the plurality of solutions, the first solution providing a first service to be executed in a cloud, the first solution comprising a specific set of operations accessible by a user;
>
> for the first solution, receive an implementation selection responsive to selection by the user of a selected implementation from a plurality of different implementations of the first solution providing the first service, the different implementations of the first solution comprising a software as a service (SaaS) implementation of the first solution, a platform as a service (PaaS) implementation of the first solution, and an infrastructure as a service (IaaS) implementation of the first solution;
>
> obtain configuration information relating to a configuration of the selected implementation; and
>
> communicate with an endpoint of the first service based on the selected implementation and the configuration information, the first service to allocate a resource of the cloud based on the configuration information and the selected implementation.

Appeal Br. i (Claims App'x).

We have reviewed the Examiner's rejection in light of Appellant's arguments presented in this appeal. Arguments which Appellant could have made but did not make in the Brief are deemed to be waived. *See* 37 C.F.R. § 41.37(c)(1)(iv)(2016).

PATENT-ELIGIBILITY REJECTION

Claims 1–12 and 16–25 are rejected under 35 U.S.C. § 101 as being directed to non-statutory subject matter. Ans. 3–6. The Examiner found that claim 1 is directed to an abstract idea, which "describe[s] the concept of providing a plurality of selections to a user for selection, configuring and providing the user's selections." *Id.* at 4; *id.* at 3–4. The Examiner stated the claim's steps are similar to *Affinity Labs v. Direct TV*, 838 F.3d 1253 (Fed. Cir. 2016) and *OIP Techs. v. Amazon.com Inc.*, 788 F.3d 1359 (Fed. Cir. 2016). The Examiner identified various elements in claim 1 as additional elements. Ans. 4–5. The Examiner further stated the additional elements of "obtaining configuration information" and "receiving information to implement the selected solution" are generic computer functions that are well-understood, routine, and conventional activities. *Id.* at 5. The Examiner concluded claim 1's additional elements are not sufficient to amount to significantly more than the judicial exception and do not improve a computer's functioning or other technology. *Id.* at 5–6.

Among other contentions, Appellant specifically points to the Specification in asserting the claims address improving a computer's functionality by enhancing selections from among multiple implementation options for a solution. Reply 4 (citing Spec. ¶ 8). Appellant further contends claim 1 concerns deploying a service in a cloud in an improved manner and does not just perform tasks related to communicating and displaying content. *Id.* at 5. Appellant also asserts claim 1's additional features are sufficient to transform the claimed subject matter into a patent-eligible application. *Id.* at 6–7, 10–11 (citing *DDR Holdings, LLC v. Hotels.com, L.P.*, 773 F.3d 1245, 1255 (Fed. Cir. 2014)).

MAIN ISSUE

Under § 101, has the Examiner erred in rejecting independent claims 1, 6, and 11 by determining that the claims are directed to judicially excepted, patent ineligible subject matter?

PRINCIPLES OF LAW

An invention is patent eligible if it claims a "new and useful process, machine, manufacture, or composition of matter." 35 U.S.C. § 101. However, the Supreme Court has long interpreted 35 U.S.C. § 101 to include implicit exceptions: "[L]aws of nature, natural phenomena, and abstract ideas" are not patentable. *Mayo Collaborative Servs. v. Prometheus Labs., Inc.*, 566 U.S. 66, 70 (2012) (brackets in original) (citing *Diamond v. Diehr*, 450 U.S. 175, 185 (1981)).

In determining whether a claim falls within an excluded category, we are guided by the Supreme Court's two-step framework, described in *Mayo* and *Alice*. *Alice Corp. v. CLS Bank Int'l*, 573 U.S. 208, 217–18 (2014) (citing *Mayo*, 566 U.S. at 75–77). In accordance with that framework, we first determine what concept the claim is "directed to." *See Alice*, 573 U.S. at 219 ("On their face, the claims before us are drawn to the concept of intermediated settlement, *i.e.*, the use of a third party to mitigate settlement risk."); *see also Bilski v. Kappos*, 561 U.S. 593, 611 (2010) ("Claims 1 and 4 in petitioners' application explain the basic concept of hedging, or protecting against risk.").

Concepts determined to be abstract ideas, and thus patent ineligible, include certain methods of organizing human activity, such as fundamental economic practices (*Alice*, 573 U.S. at 219–20; *Bilski*, 561 U.S. at 611);

mathematical formulas (*Parker v. Flook*, 437 U.S. 584, 594–95 (1978)); and mental processes (*Gottschalk v. Benson*, 409 U.S. 63, 69 (1972)). Concepts determined to be patent eligible include physical and chemical processes, such as "molding of rubber products" (*Diehr*, 450 U.S. at 193); "tanning, dyeing, making water-proof cloth, vulcanizing India rubber, smelting ores" (*id.* at 182 n.7 (quoting *Corning v. Burden*, 56 U.S. (15 How.) 252, 267–68 (185))); and manufacturing flour (*Benson*, 409 U.S. at 69 (citing *Cochrane v. Deener*, 94 U.S. 780, 785 (1876))).

If the claim is "directed to" an abstract idea, we turn to the second step of the *Alice* and *Mayo* framework, where "we must examine the elements of the claim to determine whether it contains an 'inventive concept' sufficient to 'transform' the claimed abstract idea into a patent-eligible application." *Alice*, 573 U.S. at 221 (quotation marks omitted). "A claim that recites an abstract idea must include 'additional features' to ensure 'that the [claim] is more than a drafting effort designed to monopolize the [abstract idea].'" *Id.* (quoting *Mayo*, 566 U.S. at 77). "[M]erely requir[ing] generic computer implementation[] fail[s] to transform that abstract idea into a patent-eligible invention." *Id.*

In January 2019, the USPTO published revised guidance on the application of § 101. *2019 Revised Patent Subject Matter Eligibility Guidance*, 84 Fed. Reg. 50 (Jan. 7, 2019) ("Revised Guidance"). Under the Revised Guidance, we first look to whether the claim recites:

> (1) any judicial exceptions, including certain groupings of abstract ideas (i.e., mathematical concepts, certain methods of organizing human activities such as a fundamental economic practice, or mental processes) (Revised Guidance, 84 Fed. Reg. at 52–54); and

> (2) additional elements that integrate the judicial exception into a practical application (*see* the Manual of Patent Examining Procedure (MPEP) §§ 2106.05(a)–(c), (e)–(h)) (Revised Guidance, 84 Fed. Reg. at 53–55).

Only if a claim (1) recites a judicial exception, and (2) does not integrate that exception into a practical application ("Revised Step 2A"), do we then look to whether the claim:

> (3) adds a specific limitation beyond the judicial exception that is not well-understood, routine, and conventional in the field (*see* MPEP § 2106.05(d)); or
>
> (4) simply appends well-understood, routine, and conventional activities previously known to the industry, specified at a high level of generality, to the judicial exception.

See Revised Guidance, 84 Fed. Reg. at 56 ("Step 2B").

ANALYSIS

This application has three independent claims that vary in scope: (1) claim 1 recites a "cloud orchestration system" having a processor and non-transitory storage medium with instructions executable on the processor to perform various functions; (2) claim 6 recites a "non-transitory computer readable medium" having instructions executable by a system comprising a processor to perform various functions similar to those in claim 1; and (3) claim 11 recites a method "for orchestration of a cloud comprising" various steps with similarities to claims 1 and 6. However, claim 1 recites "different implementations" of a solution include SaaS, PaaS, and IaaS, whereas claims 6 and 11 do not. Also, claims 1 and 6 recites a function that communicates with a service's endpoint to allocate cloud resources, whereas claim 11 does not. Claim 11, on the other hand, recites "instantiating . . . a first solution providing the first service," whereas claims 1 and 6 do not.

We address these claims individually, beginning with claim 11.

Claim 11 recites a "method for orchestration of a cloud" comprising:

> causing, by a system comprising a processor, presentation by a marketplace of a plurality of solutions providing different services, each solution of the plurality of solutions including a respective plurality of different implementation options;
>
> receiving, by the system, user selection of a first solution of the plurality of solutions, the first solution providing a first service to be executed in a cloud, the first solution comprising a specific set of operations accessible by a user;
>
> receiving, by the system, selection of a selected implementation for the first solution providing the first service, the selected implementation selected by the user from among a plurality of different implementations of the first solution providing the first service, the cloud configurable to provide the first solution via a plurality of different configurations of the cloud corresponding to the different implementations of the first solution;
>
> collecting, by the system, configuration information relating to the configuration of the selected implementation;
>
> instantiating, by the system, the first solution providing the first service based on the selected implementation and the configuration information; and
>
> providing, by the system, connection information associated with the first solution executed on the cloud to provide the first service.

Appeal Br. iv (Claims App'x).

Alice Step One

Claim 11 recites a method, which falls within one of the categories set forth in § 101. *See* Revised Guidance, 84 Fed. Reg. at 53–54. Despite falling within a statutory category, we must determine whether claim 11 as a whole is directed to a judicial exception, namely an abstract idea. *See Alice*, 573 U.S. at 217. To this end, we determine (1) whether claim 11 recites a judicial exception ("Revised Step 2A - Prong 1") and, if so, (2) whether the identified judicial exception is integrated into a practical application ("Revised Step 2A – Prong 2"). *See* Revised Guidance, 84 Fed. Reg. at 52–55. If both elements are satisfied, the claim is directed to a judicial exception under the first step of the *Alice* test. *See id.*

We identify claim 11's specific limitations that recite a judicial exception, and determine whether the identified limitations fall within certain subject matter groupings, namely (a) mathematical concepts (mathematical relationships, formulas, and calculations); (b) certain methods of organizing human activity (e.g., fundamental economic practices, commercial or legal interactions, managing personal behavior, or relationships or interactions between people); or (c) mental processes (e.g., concepts performed in the human mind including an observation, evaluation, judgment, or opinion). *See* Revised Guidance, 84 Fed. Reg. at 52.

When evaluating claim 11, its recited steps can be viewed as directed to selecting (1) a solution from a plurality of presented solutions to provide a service and (2) an implementation from a plurality of different implementations for the selected solution to provide a service. *See* Ans. 4 (stating the steps "describe the concept of providing a plurality of selections to a user for selection"), which fits within at least one of the above categories of the agency's guidelines as explained below.

For example, the step reciting "causing . . . presentation . . . of a plurality of solutions providing different services" information could be done by a person merely writing down and showing the solutions that provide different services—a step that can involve mere judgment and logical reasoning. *Cf. CyberSource Corp. v. Retail Decisions, Inc.*, 654 F.3d 1366, 1372–73 (Fed. Cir. 2011) (noting that a recited step that utilized a map of credit card numbers to determine the validity of a credit card transaction could be performed entirely mentally by merely using logical reasoning). Accordingly, the "causing . . . presentation" step falls squarely within the mental processes category of the agency's guidelines and, therefore, recites an abstract idea. *See* Guidance, 84 Fed. Reg. at 52 (listing exemplary mental processes include judgment).

The step reciting "receiving" a "user selection of a first solution of the plurality of solutions" in claim 11 could also be done by observing and evaluating the solutions and mentally or writing down the selected solution—a step that can involve mere observation, evaluation, and logical reasoning. *See Mortg. Grader Inc. v. First Choice Loan Servs., Inc.*, 811 F.3d 1314, 1324 (Fed. Cir. 2016) (holding a claim reciting, among other things, selecting information, upon determining a borrower wishes to obtain a loan, could be performed by humans without a computer). Similarly, the step of "receiving" a "selection of a selected implementation for the first solution providing the first service, the selected implementation selected by the user from among a plurality of different implementations of the first solution providing the first service" could be done by observing and evaluating the implementations for the solution and mentally or writing down the selected implementation. Accordingly, the "receiving" steps fall squarely within the mental processes category of the agency's guidelines and, therefore, recite an abstract idea. *See* Guidance, 84 Fed. Reg. at 52 (listing exemplary mental processes include observation and evaluation).

Finally, the additional steps "collecting" configuration information and "providing" connection information could also be done by evaluating and judging information mentally or writing down the information—steps that can involve mere evaluation and judgment. *Cf. CyberSource*, 654 F.3d at 1372–73.

Thus, although we agree with Appellant that claim 11 is not sufficiently similar to *Affinity Labs* or *OIP Techs* (*see* Reply Br. 3–6), we determine that claim 11 fall squarely within the mental processes category of the agency's guidelines and, therefore, recites an abstract idea. Claims 1 and 6 also recites similar functions to claim 11 identified above (e.g., executable instructions to cause presentation of solutions, to receive user selection from the presented solutions, to receive a selection of implementation for the selected solution from a plurality of implementations for the solution, and to obtain configuration information) and thus, the functions in claims 1 and 6 could be performed by observing, evaluating, and judging mentally or using pen and paper as explained previously. Accordingly, the "cause presentation," "receive," and "obtain" functions in claims 1 and 6 fall squarely within the mental processes category of the agency's guidelines and, therefore, recites an abstract idea. *See* Guidance, 84 Fed. Reg. at 52.

We must further determine whether the identified judicial exception for independent claims 1, 6, and 11 is integrated into a practical application ("Revised Step 2A – Prong 2"). *See* Revised Guidance, 84 Fed. Reg. at 52–55, namely whether the claim applies, relies on, or uses the abstract idea in a manner that imposes a meaningful limit on the abstract idea, such that the claim is more than a drafting effort designed to monopolize the abstract idea. *See id.* at 54–55. To this end, we (1) identify whether there are any additional recited elements beyond the judicial exception, and (2) evaluate those elements individually and collectively to determine whether they integrate the exception into a practical application. *See id.*

Starting again with independent claim 11, the recited (1) "cloud," (2) "system comprising a processor," (3) service "to be executed in a cloud," (4) first solution executed "on the cloud," and (5) step of "instantiating, by the system, the first solution providing the first service based on the selected implementation and the configuration information" are the elements beyond the abstract idea. *See* Ans. 4–5. The Examiner found these limitations are not sufficient to amount to more than the judicial exception because they perform generic computer functions. *Id.* Appellant responds that the claimed subject matter is directed to improving computer functionality (Reply Br. 4–5 (citing Spec. ¶ 8)) and rooted in computer technology to overcome a problem specifically arising in computer technology (*id.* at 10–11 (citing *DDR Holdings*, 773 F.3d at 1255). Upon review, we agree with Appellant.

Additional elements that may integrate the exception into a practical application include: (1) improving the computer itself; (2) improving another technology or technical field; (3) implementing the abstract idea in conjunction with a particular machine or manufacture that is integral to the claim; (4) transforming or reducing a particular article to a different state or thing; or (5) applying or using the abstract idea in some other meaningful way beyond generally linking the abstract idea's use to a particular technological environment, such that the claim as a whole is more than a drafting effort designed to monopolize the exception. *See* Guidance, 84 Fed. Reg. at 55 (citing MPEP §§ 2106.05(a)–(c), (e)).

Like *DDR Holdings*, 773 F.3d 1245, claim 11 does not merely recite performing the identified mental processes known from pre-cloud world. Instead, claim 11 and its recited "solution is necessarily rooted in computer technology in order to overcome a problem specifically arising in the realm of" cloud computing environment (*id.* at 1257) as Appellant explains (*see* Reply Br. 3–4, 10–11).

To illustrate, the Specification states "[c]loud providers commonly provide applications based on a single service model and restrict offerings available under other implementations." Spec. ¶ 7. Examples of a cloud service model includes a software as a service (SaaS), a platform as a service (PaaS), and an infrastructure as a service (IaaS). *Id.* ¶¶ 1, 7. The Specification explains the present invention orchestrates a cloud based on an implementation selection. *Id.* ¶ 8.

> A cloud provider can orchestrate multiple service solutions based on the implementation options available for the solution. By centralizing implementation options and associated methods of communication with the service, a cloud provider can orchestrate the multiple solutions on a single cloud. By creating a single interface to communicate with the service and configuration, the customers of cloud service providers can receive a service with a variety of implementation options and select the implementation that fits the customer's desires.

Id.

Claim 11 reflects the above-described improvement by (1) "present[ing]" the user selections, including providing a plurality of solutions and implementation options, (2) receiving the selections through a system (e.g., centralizing implementation options through an interface to communicate with the service and configuration). Claims 1 and 6 recite similar limitations. Appeal Br. i–ii (Claims App'x).

However, with exception of the recited "system comprising a processor" and the "service to be executed in a cloud" in claim 11, these features are arguably part of the identified judicial exception (e.g., part of the mental processes). Appeal Br. iv (Claims App'x). As such, the noted improvement may be considered an improvement to the identified abstract idea itself, and the additional elements (e.g., system comprising a processor) may be viewed as merely tools to implement the abstract idea and to link the judicial exception generally to a particular technical environment (e.g., to a cloud computing environment). *See Alice*, 573 U.S. at 223 (noting "a mere recitation to a generic computer cannot transform a patent-ineligible abstract idea into a patent-eligible invention" holding merely implementing a mathematical principle on a general purpose computer is a patent ineligible abstract idea), 225; *see DDR Holdings*, 773 F.3d at 1256 (noting claims directed an abstract idea of using advertising as currency applied to a particular technical environment of the Internet are claims "to nothing more than the performance of an abstract business practice on the Internet" and "are not patent-eligible") (citing *Ultramercial, Inc. v. Hulu, LLC*, 772 F.3d 709, 715–16 (Fed. Cir. 2014)).

But, claim 11 further recites the additional element of "instantiating" the selected solution by providing the service based on the selected implementation and having the first solution executed on the cloud. Appeal Br. iv (Claim App'x). This step in combination with the identified judicial exception further reflects an improvement discussed in the Specification of a cloud provider orchestrating the multiple solutions on a single cloud (e.g., instantiating *based on the selected implementation*). See Spec. ¶ 8; *see Finjan, Inc. v. Blue Coat Systems, Inc.*, 879 F.3d 1299, 1303–04 (Fed. Cir. 2018) (finding virus screening by itself is an abstract idea but the recited behavior-based virus scan "does a good deal more" and constituted an improvement in computer functionality).

For example, the Microsoft Computer Dictionary defines (1) "instantiate" as "[t]o create an instance of a class. *See also* class, instance, object (definition 2),"[3] (2) an "instance" as "[a]n object, in object-oriented programming, in relation to the class to which it belongs,"[4] (3) an "object" as "[i]n object-oriented programming, a variable comprising both routines and data that is treated as a discrete entity,"[5] and (4) a "class" as "[i]n object-oriented programming, a generalized category that describes a group of more specific items, called *objects*, that can exist within it. A class is a descriptive tool used in a program to define a set of attributes or a set of services (actions available to other parts of the program) that characterize any member (object) of the class."[6] As such, the "instantiating" step in claim 11 involves object-oriented programming and creating a generalized category object that further describes other objects, each object having a variable comprising both routines and data treated as a discrete entity, *based on the selected implementation*.

Thus, the "instantiating . . . the first solution providing the first service" step in claim 11's invention roots the claim to a computing

[3] *Instantiate*, The Microsoft Computer Dictionary 276 (5th ed. 2002).
[4] *Instance*, *id.*
[5] *Object*, *id.* at 372.
[6] *Class*, *id.* at 100.

technology (e.g., object-oriented programming). Moreover, the additional element that the "first solution [is] executed on the cloud" further roots the noted improved features to cloud-computing technology and reflects the noted improvement in cloud-computing technology in order to overcome a problem specifically arising in the realm of cloud computing environment as previously noted (*see* Spec. ¶¶ 7–8 (discussing the cloud commonly provides a single service model and restricts implementation options and this invention orchestrates multiple solutions on a single cloud)). *See DDR Holdings*, 773 F.3d at 1257; *see* Reply Br. 10–11.

As for independent claims 1 and 6, these claims recite "a first service to be executed in a cloud" but do not recite an "instantiating" function. Appeal Br. i–ii (Claims App'x). However, claim 1 recites other additional elements, namely "a cloud orchestration system," "a processor," "a non-transitory storage medium," "a first service to be executed in a cloud," "the different implementations of the first solution comprising a software as a service (SaaS) implementation of the first solution, a platform as a service (PaaS) implementation of the first solution, and an infrastructure as a service (IaaS) implementation of the first solution," "an endpoint," and "the first service to allocate a resource of the cloud based on the configuration information and the selected information." Appeal Br. i (Claim App'x).

These additional elements necessarily root claim 1's solution in computer technology to overcome the above-noted problem arising in the cloud computing and apply the identified judicial exception in some meaningful way beyond generally linking the use of the judicial exception to a particular technology. For example, claim 1 recites "communicate with an endpoint of the first service based on the selected implementation . . . , the first service to allocate a resource of the cloud based on the configuration information and the selected implementation," which roots the noted improved features to cloud-computing technology and further reflects the noted improvement by allocating resources "based on the selected implementation" to overcome a problem specifically arising in cloud computing. *See DDR Holdings*, 773 F.3d at 1257. The Specification supports this determination, indicating a resource (e.g., a container or application) is allocated to the selected solution through an endpoint (e.g., an interface) based on the selected implementation. *See* Spec. ¶¶ 9, 20, 29, 37; *see* Reply Br. 11. Claim 6 also recites similar limitations. Appeal Br. ii (Claims App'x).

Additionally, claim 1 recites the recited "different implementations of the first solution" used by the first service "to allocate a resource" comprising SaaS, PaaS, and IaaS implementations. These additional elements that implement the selected solution using SaaS, PaaS, and IaaS, also directs claim 1 to a specific implementation of a solution to the above-identified problem in the cloud computing arts. *See Enfish, LLC v. Microsoft Corp.*, 22 F.3d 1327, 1339 (Fed. Cir. 2016); *see also Finjan*, 879 F.3d at 1305.

For the above reasons, the additional recited elements in claims 1, 6, and 11—considered individually and as an ordered combination—integrate the identified judicial exception into a practical application.

Alice Step Two

Although we determined above that the additional recited elements in claims 1, 6, and 11—considered individually and as an ordered combination—integrate the identified judicial exception into a practical application, we add the following to support further that claims 1, 6, and 11 is directed to patent eligible subject matter. *DDR Holdings* explains, "claims [can] recite an invention that is not merely the routine or conventional use of the Internet" "when the limitations of [a] patent's asserted claims are taken together as an ordered combination." *DDR Holdings*, 773 F.3d at 1259. For example, the claims in *DDR Holdings* "specify how interactions with the Internet are manipulated to yield a desired result—a result that overrides the routine and conventional sequence of events ordinarily triggered by the click of a hyperlink." *Id.* at 1258. These claims in *DDR Holdings* were found to "recite a specific way to automate the creation of a composite web page by an 'outsource provider' that incorporates elements from multiple sources in order to solve a problem faced by websites on the Internet," amounting to an inventive concept for resolving a problem particular to the Internet. *Id.* at 1259.

Similarly here, the additional elements in claims 1, 6, and 11 taken together as an ordered combination amount to an inventive concept resolving a problem particular to cloud computing. As explained above, the Specification states cloud provides "*commonly* provide applications based on a single service model." Spec. ¶ 7 (emphasis added). Claim 11, on the other hand, provides "[a] cloud provider [that] can orchestrate multiple service solutions based on the implementations options available for the solution" and "multiple solutions on a single cloud." *Id.* ¶ 8. Claim 11's additional elements taken together as an ordered combination therefore recite an invention that is not merely a routine and conventional use in cloud computing. The additional elements in claim 11 also include specific interactions with the cloud that permit a user to select a particular solution from a plurality of solutions and a particular implementation from a plurality of implementations for the selected solution to yield a desired result that fits a customer's desires (*see id.*), which differs from the common manner of providing service solutions on the cloud as the Specification explains. We therefore determine that claim 11's additional elements collectively amount to an inventive concept for resolving a problem particular to cloud computing. *See DDR Holdings*, 773 F.3d at 1259. Claim 1's and claim 6's additional elements, likewise, taken as an ordered combination recite an invention that is beyond the routine and conventional use of cloud computing for similar reasons to claim 11.

We further note "an inventive concept can be found in the non-conventional and non-generic arrangement of known, conventional pieces." *Bascom Global Internet Srvs., Inc. v. AT&T Mobility LLC*, 827 F.3d 1341, 1350. Based on the above discussion related to improving on the common technique for providing services in cloud computing, the additional elements in these claims can be also be viewed as going beyond well-understood, routine, and conventional cloud computing activities.[7] *See id.* at 1348. As such, the additional elements in claims 1, 6, and 11 as an ordered combination can also be viewed as amounting to a non-conventional and non-generic arrangement of known, conventional pieces, which add significantly more than the abstract idea to provide an inventive concept

[7] That Odenheimer may teach centralizing selected implementations to orchestrate multiple solutions on a cloud, as discussed below, does not belie this point. *See Return Mail, Inc. v. United States Postal Service*, 868 F.3d 1350, 1370 (Fed. Cir. 2017) (stating "§ 101 subject-matter eligibility is a requirement separate from other patentability inquiries.").

under *Alice/Mayo* step two. *See Alice*, 573 U.S. at 221; *see also* Guidance, 84 Fed. Reg. at 56.

Accordingly, we are persuaded that the Examiner erred in rejecting (1) claims 1, 6, and 11, and (2) dependent claims 2–5, 7–10, 12, and 16–25 under 35 U.S.C. § 101.

OBVIOUSNESS REJECTION OVER ODENHEIMER AND KAMPAS

Claims 1–6, 11, 12 and 16–19

Claims 1–12 and 16–23 are rejected under 35 U.S.C. 103 as being unpatentable over Odenheimer (US 2012/0054626 Al, published Mar. 1, 2012) and Kampas (US 2012/0124211, published May 12, 2012). Final Act. 3–16. Appellant argues claims 1–5 and 16 as a group and repeats the arguments related to claim 1 for claims 6, 11, 12, and 17–19. Appeal Br. 6–11. We select claim 1 as representative of claims 1–6, 11, 12, and 16–19. *See* 37 C.F.R. § 41.37(c)(1)(iv).

Regarding claim 1, the Examiner found Odenheimer teaches many of claim 1's limitations, including "each solution . . . including a respective plurality of different implementation options." Final Act. 3–5 (citing Odenheimer ¶¶ 20–21, 29–31, 40, 45, 49, Fig. 4); Ans. 8–9 (further citing Odenheimer ¶¶ 4, 25). The Examiner turned to Kampas in combination with Odenheimer to teach or suggest the "communicate" limitation in claim 1. Final Act. 5 (citing Kampas ¶¶ 11–12, 66, Fig. 6).

Appellant argues Odenheimer teaches IaaS, PaaS, and SaaS provides implementations of different cloud "services (solutions)" rather than different implementations of the first solution as recited. Appeal Br. 7–9 (citing Odenheimer ¶¶ 20–21, 29–31); Reply Br. 13–15. Appellant also argues Odhenheimer's discussion of service 180, which provides a PaaS or IaaS, addresses a cloud service provider shown in Figure 1 providing two different services. *Id.* at 9 (citing Odenheimer ¶¶ 29, 30, and 40). Appellant further contends Kampas does not remedy the alleged deficiencies in Odenheimer. *Id.*

MAIN ISSUE

Under § 103, has the Examiner erred in rejecting claim 1 by finding that Odenheimer and Kampas would have taught or suggested "a plurality of solutions providing different services, each solution . . . including a respective plurality of different implementation options"?

ANALYSIS

We begin by construing the disputed limitation of claim 1 reciting "a plurality of solutions providing different services, each solution . . . including a respective plurality of different implementations . . . ." Appeal Br. i (Claims App'x). The Examiner "interpreted the 'solution' as a cloud service that can be implemented in cloud platforms such as IaaS, PaaS, and SaaS to provide cloud services." Final Act. 3; *see* Ans. 8. The Examiner *further* provided examples of "services" to include "developing a software, SAP business byDesign etc." and "these different services can be provided via PaaS, SaaS, and IaaS platforms." *Id.* (citing Odenheimer ¶¶ 002–0031). Although Appellant acknowledges this interpretation, Appellant does not address whether the recited "service" includes developing software or using an application, such as SAP's Business byDesign, as the Examiner found. *See* Appeal Br. 8; *see* Reply Br. 13–14.

Based on the record, we find the Examiner's position consistent with the Specification. The Specification states "[a] service . . . is any appropriate supplying of communication, transmissions, software, storage, or any other product, resource, or activity that is capable of executing on a network of electronic devices." Spec. ¶ 6. Thus, the phrase "a first service" in claim 1 construed broadly, but reasonably in light of the disclosure encompasses many hardware and software functions, including communication, transmission, and storage, as well as providing software or a product/activity executable in a device network. *See id.* Additionally, the broadest reasonable construction of "a plurality of solutions providing different services," including "the first solution providing a first service," in claim 1 are solutions that provide one of the above-noted features (e.g., communication, software, or a product/activity executable in a device network). *See id.* ¶¶ 6, 9–10; *see In re Am. Acad. of Sci. Tech Ctr.*, 367 F.3d 1359, 1364 (Fed. Cir. 2004) (citation omitted).

Turning to Odenheimer, this reference teaches or suggests the recited "plurality of solutions providing different services, each solution . . . including a respective plurality of different implementation options." The Examiner relies on several passages in Odenheimer to teach this feature. Final Act. 4 (citing Odenheimer ¶¶ 20–21, 29–31, 40); Ans. 8–9 (further citing Odenheimer ¶¶ 4, 25). For example, Odenheimer teaches a cloud computing, including "enabl[ing] a user at a client computer to access computing resources provided by a service," the computer resources "includ[ing] hardware (e.g., computing capability, storage, etc.), software (e.g., operating systems, end-user software applications, etc.), and a combination of both." Odenheimer ¶ 4. Further examples of services provided are described in Odenheimer. *Id.* ¶¶ 29 (describing selecting a development system used to develop software systems, such as a Linux system), 31 (describing selecting an end-user software application, such as SAP Business byDesign), 39 (describing developing end-user applications), 40 (describing developing an application). As such, Odenheimer teaches and suggests a system where a user can select: (1) accessing hardware (e.g., storage), (2) accessing software (e.g., end-user applications), and (3) developing a software system or an end-user application to name a few. Odenheimer therefore at least suggests a cloud computing system that enables a user to select "a plurality of solutions providing different services" as claim 1 broadly recites.

Likewise, Odenheimer's cloud service provider 180, discussed in cited paragraphs 20, 21, 25, and 40 (Final Act. 4; Ans. 8–9), enables a user to select and access storage, end-user applications, and other products executable in a device network (i.e., "a plurality of solutions providing different services").

Odenheimer further teaches and suggests a solution to a service includes "different implementation options," including SaaS, PaaS, and IaaS, as recited in claim 1. *See* Final Act. 4 (citing Odenheimer ¶¶ 20–21, 29–31, 40). In particular, Odenheimer teaches enabling selection of IaaS, PaaS, and/or SaaS through provisioning layer 160 and cloud service provider 180. Odenheimer ¶¶ 20, 25, 34, 36. As an example, Odenheimer describes a service of hosting software that includes selecting a hardware/software stack, including selecting or implementing PaaS only. *Id.* ¶ 30. Similarly, Odenheimer describes hosting/maintaining software, such as a company application (e.g., a service), by implementing both PaaS and IaaS to increase database and operating system flexibility due to the limitations of given PaaS stack (e.g., hardware/software stack). *Id.* ¶ 40; *see also id.* ¶ 30 (discussing selecting PaaS and "an IaaS package" when hosting software). Odenheimer thus suggests two different implementations (e.g., PaaS only and combined PaaS/IaaS) for a service (e.g., hosting/maintaining software). Odenheimer suggests other services (e.g., hosting other software or developing end-user applications) can be provided using or implementing pure IaaS, pure PaaS, PaaS and IaaS, pure SaaS, SaaS and PaaS, and SaaS, PaaS, and IaaS. *See id.* ¶ 25, 29, 31, 36, 39, 40.

Thus, although provider 180 provides different services as discussed in Odenheimer's paragraphs 29 and 30, the above illustration demonstrates the cited passages in Odenheimer by the Examiner, when viewed as a whole, do not only describe different implementations of different services as alleged. *See* Appeal Br. 8–9; *see* Reply Br. 13–16. Rather, Odenheimer at least suggests "a plurality of solutions providing different services, each solution . . . including a respective plurality of different implementation options," as claim 1 recites.

Lastly, Appellant asserts the Examiner erred in finding Odenheimer teaches the following limitations

> cause presentation, by a marketplace, of a plurality of solutions providing different services, each solution of the plurality of solutions including a respective plurality of different implementation options;
>
> receive user selection of a first solution of the plurality of solutions, the first solution providing a first service to be executed in a cloud, the first solution comprising a specific set of operations accessible by a user;

> for the first solution, receive an implementation selection responsive to selection by the user of a selected implementation from a plurality of different implementations of the first solution providing the first service, the different implementations of the first solution comprising a software as a service (SaaS) implementation of the first solution, a platform as a service (PaaS) implementation of the first solution, and an infrastructure as a service (IaaS) implementation of the first solution.

in claim 1. Appeal Br. 6–7; Reply Br. 12. Except as noted above, Appellant does not address the above recitations with any specificity, and this mere assertion is not considered a separate argument for patentability. *See In re Lovin*, 652 F.3d 1349, 1357 (Fed. Cir. 2011).

For the foregoing reasons, Appellant has not persuaded us of error in the rejection of independent claim 1 and claims 2–6, 11, 12, and 16–19, which are not argued separately.

Claims 7–9

Claim 7 depends from independent claim 6 and further recites "the instructions are executable by the system to: identify the endpoint to communicate with based on information collected, the information collected including information of the selected implementation and the configuration information." Appeal Br. iii (Claims App'x). Claims 8 and 9 depend ultimately from claim 7 and are not separately argued. *Id.* at 11–12. We select claim 7 as representative. *See* 37 C.F.R. § 41.37(c)(1)(iv).

The Examiner relied on Odenheimer to teach this limitation. Final Act. 9 (citing Odenheimer ¶ 38). Appellant argues the Examiner's rejection does not address claim 7 and more specifically, Odenheimer's performing a temporary task through virtual machine access does not relate to claim 7's language. Appeal Br. 11–12; Reply Br. 16–18. Appellant further asserts Kampas does not remedy the alleged deficiencies in Odenheimer. Appeal Br. 12.

We are not persuaded. Notably, claim 7 depends from claim 6, and the claim 6's rejection relies on Kampas in combination with Odenheimer to teach the limitation "communicate with an endpoint of the first service to allocate a resource of the cloud based on the selected implementation and the configuration information." Final Act. 8–9 (citing Kampas ¶¶ 11–12, 66, Fig. 6); *see* Ans. 10 (noting Kampas teaches the "communicating with the endpoint" feature in claim 6). As such, the rejection as a whole does not rely solely on Odenheimer to teach communicating with the endpoint based on collected information, including implementation and configuration information.

Turning to Kampas, this reference teaches accessing resources through a cross-platform environment, such as a portal. Kampas ¶ 66. Thus, Kampas teaches communicating with a web-page portal and at least suggests the portal must be identified in some manner prior to communication (e.g., "identify the endpoint to communicate with" as recited in claim 7). *See id.* Moreover, the Specification states an endpoint includes "a server device, a server device, a virtual machine, an application, an interface, and/or similar entity capable of receiving the information used to configure the cloud based on the selected implementation." Spec. ¶ 19. Given the breadth of the term "endpoint," we find Kampas's portal, which can be construed as an interface, reasonably maps to an endpoint as recited in claims 6 and 7.

Also, as the Examiner indicates, Odenheimer suggests identifying an endpoint to communicate with based on the collected information. *See* Final Act. 9; *see* Ans. 10. That is, paragraph 38 teaches accessing a virtual machine, which is an exemplary endpoint in the Specification. Odenheimer ¶ 38; Spec. ¶ 19. And similar to our explanation related to Kampas, this passage in Odenheimer teaches accessing (e.g., communicating with) one or more virtual machines and therefore at least suggests the virtual machine(s) must be identified in some manner prior to communication (e.g., "identify the endpoint to communicate with" as recited in claim 7). We thus disagree with Appellant that this passage does not suggest "the subject matter that is specifically recited in claim 7." Reply Br. 17.

For the foregoing reasons, Appellant has not persuaded us of error in the rejection of claim 7 and claims 8 and 9, which are not argued separately.

Claim 10

Claim 10 indirectly depends from claim 6 and further recites "the configuration payload comprises a virtual machine image based on the configuration information." Appeal Br. iii (Claims App'x). Claim 9, from which claim 10 depends directly, introduces "a configuration payload." *Id.* The Examiner relied on Odenheimer to teach the features in claims 9 and 10. Final Act. 10 (citing Odenheimer ¶ 21).

Appellant argues paragraph 21 of Odenheimer does not teach the metadata comprises a virtual machine image as recited and Kampas does not remedy this deficiency. Appeal Br. 13. We agree that Odenheimer's paragraph 21 does not discuss a virtual machine image. See Odenheimer ¶ 21.

In the Response to Argument section of the Examiner's Answer, the Examiner relied on new passages in Odenheimer. Ans. 11 (further Odenheimer ¶¶ 16, 45). More specifically, the Examiner stated Odenheimer teaches collecting configuration and deployment information (e.g., metadata) to configure the selected implementation/option. *Id.* (citing Odenheimer ¶¶ 21, 45). The Examiner further explained Odenheimer teaches IaaS can be implemented by configuring a virtual machine. *Id.* (citing Odenheimer ¶ 16). The Examiner concluded "it would be obvious to have [a] virtual machine image as metadata or configuration information for selected implementation as IaaS." *Id.*

Appellant responds, stating Odenheimer's paragraph 16 "merely indicates that a service can be implemented using a virtual machine. However, ¶ [0016] does not state that an image of such a virtual machine is included in a configuration payload that is sent to an endpoint." Reply Br. 19. We agree with Appellant.

Odenheimer teaches metadata 162 may include information related to provisioning IaaS as well as include information associated with the configuration and deployment. Odenheimer ¶¶ 21, 45. Odenheimer further teaches that IaaS delivers a service as a virtualized environment implemented on a virtual machine (e.g., a VMware stack, a XEN stack. A Solaris-based stack, or like hypervisor being offered). *Id.* ¶ 16. Collectively, these teachings at best suggest metadata may include information related to implementing the service on a virtual machine. *See id.* ¶¶ 16, 21, 45. However, there is insufficient discussion in Odenheimer regarding what form the metadata may take, and thus, on the record, Odenheimer fails to teach or suggest claim 10's "configuration payload comprises a virtual machine image based on the configuration information." *See* Reply Br. 19 (arguing paragraph "[0016] does not state that an image of such a virtual machine is included in a configuration payload that is sent to an endpoint.").

Accordingly, Appellant has persuaded us of error in the rejection of claim 10.

Claims 20–23

Claim 20 depends from independent claim 11 and further recites

> wherein a first implementation of the plurality of different implementations of the first solution comprises a first configuration of the different configurations of virtual machines, and a second implementation of the plurality of different implementations of the first solution comprises a second, different configuration of the different configurations of virtual machines.

Appeal Br. v (Claims App'x).

Appellant groups claim 22 with claim 20 and argues claims 21 and 23 are allowable for the same reasons as claims 20 and 1[8] respectively. Appeal Br. 13–14. Claims 21 and 22 recite limitations similar in scope to claim 20, and claim 23 depends from 22. Appeal Br. v–vi (Claim App'x). We thus treat claims 20–23 as a group and select claim 20 as representative. *See* 37 C.F.R. § 41.37(c)(1)(iv).

The Examiner relied on Odenheimer to teach the features in claim 20, noting paragraph 16 teaches IaaS provides a virtual machine to the end user once configured with associated metadata and paragraph 17 teaches PaaS provides a platform as service to allow users to develop their applications.

[8] Notably, claim 23 depends from 22, which depends from claim 6 and not claim 1. Appeal Br. vi (Claim App'x).

Final Act. 14 (citing Odenheimer ¶¶ 16–18, 40). The Examiner elaborated in the Answer, stating there are at least two options in Odenheimer to be implemented as virtual machine configurations—one being IaaS implemented as a virtual machine and the other being PaaS implemented as a "physical computer functionality as virtual machines." Ans. 12 (citing Odenheimer ¶¶ 16–17).

Appellant argues Odenheimer does not teach PaaS or SaaS includes a configuration of virtual machines. Appeal Br. 14; Reply Br. 20–21. We agree. Odenheimer discloses implementing one service/solution as a virtualized environment on a virtual machine using an IaaS but does not discuss using PaaS or SaaS.[9] We further note that paragraph 17 does not teach virtual machines, a physical computer functionality as virtual machines in PaaS, or a different configuration of a virtual machine than that addressed in paragraph 16. Reply Br. 21 (stating "[t]he Examiner concluded that this is a 'physical computer functionality as virtual machines.' There is . . . no support for this assertion, as ¶ [0017] of Odenheimer makes no mention of virtual machines.").

The Examiner relied on additional passages in the Examiner's Answer, asserting the teachings in Odenheimer "disclose provisioning SaaS, PaaS, and IaaS to provide a computer functionality (e.g., hardware or software)." Ans. 12 (Odenheimer ¶¶ 4, 15). Although discussing computing resources as hardware, software, or some combination of both (Odenheimer

[9] Notably, paragraph 16 discusses IaaS may be provided as a virtual machine (e.g., VMware stack, a XEN stack, a Solaris-based provider) as well as implementing "a plurality of virtual machines." Odenheimer ¶ 16. The Examiner does not discuss these alternative virtual machines or the plurality of virtual machines when rejecting claim 20. Final Act. 14; Ans. 12.

¶ 4) and discussing cloud computing system using SaaS, IaaS, and PaaS (*id.* at ¶ 15), we agree with Appellant that these cited passages do not disclose virtual machines or more specifically,

> a first implementation of the plurality of different implementations of the first solution comprises a first configuration of the different configurations of virtual machines, and a second implementation of the plurality of different implementations of the first solution comprises a second, different configuration of the different configurations of virtual machines.

as claim 20 recites.

Given the record, we are constrained to conclude the Examiner erred in the rejecting of (1) claim 20, (2) claims 21 and 22, which are commensurate in scope with claim 20, and (3) claim 23, which depends from claim 22.

OBVIOUSNESS REJECTION OVER ODENHEIMER, KAMPAS, AND REDDY

Claims 24 and 25 are rejected under 35 U.S.C. 103 as being unpatentable over Odenheimer, Kampas, and Reedy (US 2012/0054626 Al, published Mar. 1, 2012) and Kampas (US 2013/0254897 Al, published September 26, 2013). Final Act. 16–17. This rejection is not separately argued, except to assert base claims 1 and 6 are allowable and Reedy does not overcome the alleged deficiencies of Odenheimer and Kampas. Appeal Br. 16. We are not persuaded of error for reasons previously presented when addressing claims 1 and 6, for which claims 24 and 25 respectively depend. Appeal Br. vi (Claims App'x).

DECISION

In summary:

Claims Rejected	35 U.S.C. §	Reference(s)/Basis	Affirmed	Reversed
1–12, 16–25	101	Eligibility		1–12, 16–25
1–12, 16–23	103	Odenheimer, Kampas	1–9, 11, 12, 16–19	10, 20–23
24–25	103	Odenheimer, Kampas, Reedy	24–25	
Overall Outcome			1–9, 11, 12, 16–19, 24–25	10, 20–23

AFFIRMED-IN-PART

Save Form

Notice of References Cited	Application/Control No. 14/170,404	Applicant(s)/Patent Under Patent Appeal No. 2018-006424	
	Examiner	Art Unit 2454	Page 1 of 1

U.S. PATENT DOCUMENTS

*		Document Number Country Code-Number-Kind Code	Date MM-YYYY	Name	Classification
	A	US-			
	B	US-			
	C	US-			
	D	US-			
	E	US-			
	F	US-			
	G	US-			
	H	US-			
	I	US-			
	J	US-			
	K	US-			
	L	US-			
	M	US-			

FOREIGN PATENT DOCUMENTS

*		Document Number Country Code-Number-Kind Code	Date MM-YYYY	Country	Name	Classification
	N					
	O					
	P					
	Q					
	R					
	S					
	T					

NON-PATENT DOCUMENTS

*		Include as applicable: Author, Title Date, Publisher, Edition or Volume, Pertinent Pages)
	U	Microsoft Computer Dictionary - instantiate, instance, class, object
	V	
	W	
	X	

*A copy of this reference is not being furnished with this Office action. (See MPEP § 707.05(a).)
Dates in MM-YYYY format are publication dates. Classifications may be US or foreign.

Microsoft

Microsoft Computer Dictionary

Fifth Edition

PUBLISHED BY
Microsoft Press
A Division of Microsoft Corporation
One Microsoft Way
Redmond, Washington 98052-6399

Library of Congress Control Number: 2002019714

Acquisitions Editor: Alex Blanton
Project Editor: Sandra Haynes

Body Part No. X08-41929

voice/circuit-switched data, B channel, ISDN. *Compare* circuit-switched data.

circuit switching *n.* A method of opening communications lines, as through the telephone system, by creating a physical link between the initiating and receiving parties. In circuit switching, the connection is made at a switching center, which physically connects the two parties and maintains an open line between them for as long as needed. Circuit switching is typically used on the dial-up telepbone network, and it is also used on a smaller scale in privately maintained communications networks. Unlike other methods of transmission, such as packet switching, it requires the link to be established before any communication can take place. *Compare* message switching, packet switching.

circular list *n.* A linked or chained list in which processing continues through all items, as in a ring, and returns to the starting point, no matter where that point is located in the list. *See also* linked list.

CIS *n.* **1.** Acronym for CompuServe Information Service *See* CompuServe. **2.** Short for contact image sensor. A light-sensitive mechanism used in scanners and fax machines. A CIS scanner reflects light from a row of light-emitting diodes (LEDs) onto a document or other object and converts the reflected light to digital images. CIS sensors are smaller and lighter than the charge-coupled devices (CCDs) traditionally used in scanners, but the image quality they produce is not as good as the image quality produced by CCDs. *See also* light-emitting diode, scanner. *Compare* charge-coupled device.

CISC *n.* Acronym for complex instruction set computing. The implementation of complex instructions in a microprocessor design so that they can be invoked at the assembly language level. The instructions can be very powerful, allowing for complicated and flexible ways of calculating such elements as memory addresses. All this complexity, however, usually requires many clock cycles to execute each instruction. *Compare* RISC.

CIX *n.* *See* Commercial Internet Exchange.

CKO *n.* Acronym for Chief Knowledge Officer. A corporate executive in charge of management and distribution of all the business and technical knowledge of a company. The CKO maximizes the value of stored knowledge by ensuring that employees have access, and by avoiding knowledge loss caused by technology-based changes and upgrades in databases and other storage.

ClariNet *n.* A commercial service that distributes news articles from United Press International (UPI) and other news agencies in newsgroups that are part of the clari. hierarchy. Unlike most other newsgroups, access to the clari. newsgroups is restricted to Internet service providers who pay a subscription fee to ClariNet.

clari. newsgroups *n.* Internet newsgroups maintained by ClariNet Communications, Inc. ClariNet newsgroups contain news articles obtained from the Reuters and United Press International wire services, SportsTicker, Commerce Business Daily, and other sources. Unlike most other newsgroups, ClariNet groups are only accessible through Internet service providers who purchase the service. *See also* ClariNet, ISP, newsgroup.

ClarisWorks *n.* *See* AppleWorks.

class *n.* **1.** In object-oriented programming, a generalized category that describes a group of more specific items, called *objects*, that can exist within it. A class is a descriptive tool used in a program to define a set of attributes or a set of services (actions available to other parts of the program) that characterize any member (object) of the class. Program classes are comparable in concept to the categories that people use to organize information about their world, such as *animal*, *vegetable*, and *mineral*, that define the types of entities they include and the ways those entities behave. The definition of classes in object-oriented programming is comparable to the definition of types in languages such as C and Pascal. *See also* object-oriented programming. **2.** For hardware, the method for grouping particular types of devices and buses according to the basic ways that they can be installed and managed by the operating system. The hardware tree is organized by device class, and Windows uses class installers to install drivers for all hardware classes.

Class A IP address *n.* A unicast IP address that ranges from 1.0.0.1 through 126.255.255.254. The first octet indicates the network, and the last three octets indicate the host on the network. *See also* Class B IP address, Class C IP address, IP address classes.

Class A network *n.* An Internet network that can define a maximum of 16,777,215 hosts. Class A networks use the first byte of an IP address to designate the network, with the first (high-order) bit set to 0. The host is designated by the last 3 bytes. Class A addressing currently allows for a maximum of 128 networks. Class A networks are best suited for sites with few networks but numerous hosts and

insider attack *n.* An attack on a network or system carried out by an individual associated with the hacked system. Insider attacks are typically the work of current or former employees of a company or organization who have knowledge of passwords and network vulnerabilities. *Compare* intruder attack.

Ins key *n. See* Insert key.

install *vb.* To set in place and prepare for operation. Operating systems and application programs commonly include a disk-based installation, or setup, program that does most of the work of preparing the program to work with the computer, printer, and other devices. Often such a program can check for devices attached to the system, request the user to choose from sets of options, create a place for the program on the hard disk, and modify system startup files as necessary.

installable device driver *n.* A device driver that can be embedded within an operating system, usually in order to override an existing, less-functional service.

Installable File System Manager *n.* In Windows 9x and Windows 2000, the part of the file system architecture responsible for arbitrating access to the different file system components. *Acronym:* IFS.

installation program *n.* A program whose function is to install another program, either on a storage medium or in memory. An installation program, also called a setup program, might be used to guide a user through the often complex process of setting up an application for a particular combination of machine, printer, and monitor.

Installer *n.* A program, provided with the Apple Macintosh operating system, that allows the user to install system upgrades and make bootable (system) disks.

instance *n.* An object, in object-oriented programming, in relation to the class to which it belongs. For example, an object *myList* that belongs to a class *List* is an instance of the class *List*. *See also* class, instance variable, instantiate, object (definition 2).

instance variable *n.* A variable associated with an instance of a class (an object). If a class defines a certain variable, each instance of the class has its own copy of that variable. *See also* class, instance, object (definition 2), object-oriented programming.

instantiate *vb.* To create an instance of a class. *See also* class, instance, object (definition 2).

instant messaging *n.* A service that alerts users when friends or colleagues are on line and allows them to communicate with each other in real time through private online chat areas. With instant messaging, a user creates a list of other users with whom he or she wishes to communicate; when a user from his or her list is on line, the service alerts the user and enables immediate contact with the other user. While instant messaging has primarily been a proprietary service offered by Internet service providers such as AOL and MSN, businesses are starting to employ instant messaging to increase employee efficiency and make expertise more readily available to employees.

Institute of Electrical and Electronics Engineers *n. See* IEEE.

instruction *n.* An action statement in any computer language, most often in machine or assembly language. Most programs consist of two types of statements: declarations and instructions. *See also* declaration, statement.

instruction code *n. See* operation code.

instruction counter *n. See* instruction register.

instruction cycle *n.* The cycle in which a processor retrieves an instruction from memory, decodes it, and carries it out. The time required for an instruction cycle is the sum of the instruction (fetch) time and the execution (translate and execute) time and is measured by the number of clock ticks (pulses of a processor's internal timer) consumed.

instruction mix *n.* The assortment of types of instructions contained in a program, such as assignment instructions, mathematical instructions (floating-point or integer), control instructions, and indexing instructions. Knowledge of instruction mixes is important to designers of CPUs because it tells them which instructions should be shortened to yield the greatest speed, and to designers of benchmarks because it enables them to make the benchmarks relevant to real tasks.

instruction pointer *n. See* program counter.

instruction register *n.* A register in a central processing unit that holds the address of the next instruction to be executed.

instruction set *n.* The set of machine instructions that a processor recognizes and can execute. *See also* assembler, microcode.

O

OAGI *n.* Acronym for Open Applications Group, Inc. A nonprofit consortium of software vendors and businesses created to develop and define XML-based interoperability specifications and standards among enterprise-scale applications. The OAGI was formed in 1995 by a small number of business enterprise software companies and organizations and has grown to more than sixty member companies.

OAGIS *n.* Acronym for Open Applications Group Integration Specification. A set of XML-based specifications and standards designed to promote B2B e-commerce by providing interoperability between enterprise-scale applications and between companies. OAGIS includes business document specifications and definitions, business process scenarios, and templates for business forms such as invoices and requisitions. OAGIS is overseen by the Open Applications Group, Inc., a nonprofit consortium of software vendors and customers. *See also* OAGI.

OASIS *n.* Acronym for Organization for the Advancement of Structured Information Standards. A consortium of technology companies formed to develop guidelines for use of XML (Extensible Markup Language) and related information standards.

Oberon *n.* An extensible object-oriented language based on Modula-2, whose later versions support the .NET Framework. *Also called:* Active Oberon for .NET.

object *n.* **1.** Short for object code (machine-readable code). **2.** In object-oriented programming, a variable comprising both routines and data that is treated as a discrete entity. *See also* abstract data type, module (definition 1), object-oriented programming. **3.** In graphics, a distinct entity. For example, a bouncing ball might be an object in a graphics program. **4.** A single, runtime instance of object type that the operating system defines. Objects visible in user mode include event, file, I/O completion port, key, object directory, port, process, section, semaphore, symbolic link, thread, timer, and token objects. Many user-mode objects are implemented through the use of a corresponding kernel-mode object. Kernel-mode-only objects include adapter, APC, controller, device, device queue, DPC, driver, interrupt, mutex, and stream file objects.

object code *n.* The code, generated by a compiler or an assembler, that was translated from the source code of a program. The term most commonly refers to machine code that can be directly executed by the system's central processing unit (CPU), but it can also be assembly language source code or a variation of machine code. *See also* central processing unit.

object computer *n.* The computer that is the target of a specific communications attempt.

object database *n.* *See* object-oriented database.

Object Database Management Group *n.* An organization that promotes standards for object databases and defines interfaces to object databases. *Acronym:* ODMG. *See also* OMG.

object file *n.* A file containing object code, usually the output of a compiler or an assembler and the input for a linker. *See also* assembler, compiler (definition 2), linker, object code.

Objective-C *n.* An object-oriented version of the C language developed in 1984 by Brad Cox. It is most widely known for being the standard development language for the NeXT operating system. *See also* object-oriented programming.

object linking and embedding *n.* *See* OLE.

Object Management Architecture *n.* *See* OMA.

Object Management Group *n.* *See* OMG.

object model *n.* **1.** The structural foundation for an object-oriented language, such as C++. This foundation includes such principles as abstraction, concurrency, encapsulation, hierarchy, persistence, polymorphism, and typing. *See also* abstract data type, object (definition 2), object-oriented programming, polymorphism. **2.** The structural foundation for an object-oriented design. *See also* object-oriented design. **3.** The structural foundation for an object-oriented application.