Internal distribution code:

(A) [-] Publication in OJ
(B) [-] To Chairmen and Members
(C) [-] To Chairmen
(D) [X] No distribution

Datasheet for the decision of 29 June 2017

Case Number: T 1090/12 - 3.5.06

Application Number: 02737140.0

Publication Number: 1405172

IPC: G06F9/00, G06F9/46

Language of the proceedings: EN

Title of invention:
SYSTEM AND METHOD FOR SOFTWARE COMPONENT PLUG-IN FRAMEWORK

Applicant:
Oracle International Corporation

Headword:
Plugin framework/ORACLE

Relevant legal provisions:
EPC 1973 Art. 84, 111(1), 112(1)(a), 56, 23(3), 164(1)

Keyword:

Decisions cited:
T 0423/11, T 0458/07, R 0020/11

This datasheet is not part of the Decision.
It can be changed at any time and without notice.

Catchword:

This datasheet is not part of the Decision.
It can be changed at any time and without notice.

Europäisches Patentamt
European Patent Office
Office européen des brevets

Beschwerdekammern
Boards of Appeal
Chambres de recours

European Patent Office
D-80298 MUNICH
GERMANY
Tel. +49 (0) 89 2399-0
Fax +49 (0) 89 2399-4465

Case Number: T 1090/12 - 3.5.06

D E C I S I O N
of Technical Board of Appeal 3.5.06
of 29 June 2017

Appellant: (Applicant)	Oracle International Corporation M/S 50P7 500 Oracle Parkway Redwood Shores, CA 94065 (US)
Representative:	D Young & Co LLP 120 Holborn London EC1N 2DY (GB)
Decision under appeal:	**Decision of the Examining Division of the European Patent Office posted on 6 December 2011 refusing European patent application No. 02737140.0 pursuant to Article 97(2) EPC.**

Composition of the Board:

Chairman	W. Sekretaruk
Members:	S. Krischer
	A. Teale

Summary of Facts and Submissions

I. The appeal is directed against the decision of the examining division, dated 6 December 2011, to refuse application No. 02737140.0 for lack of clarity. As "obiter dicta", objections concerning lack of inventive step over D3 in combination with D5 were raised.

D3 R.J. Stets et al.: "Component - based APIs for Versioning and Distributed Applications", July 1999, pages 54-61, retrieved from the Internet on 29 January 2009, http://ftp.cs.rochester.edu/u/scott/papers/1999_IEEEC_COP.pdf.

D5 P.E. Chung et al.: "DCOM and CORBA Side by Side, Step by Step, and Layer by Layer", 3 September 1997, retrieved from the Internet on 29 January 2009, http://research.microsoft.com/en -us/um/people/ymwang/papers/ HTML/DCOMnCORBA/S.html.

II. A notice of appeal was received on 23 January 2012. The appeal fee was paid on the same day. A statement of grounds of appeal was received on 21 March 2012. The main and the auxiliary request specified in the decision were maintained. Oral proceedings were conditionally requested.

III. In its summons to oral proceedings, the board gave reasons for its preliminary opinion that claim 1 of both requests lacked an inventive step over D3.

IV. In a letter dated 26 May 2017, the appellant filed a second auxiliary request.

V. Oral proceedings were held on 29 June 2017. At the end of the oral proceedings, the board announced its decision.

VI. The appellant requests (see letter dated 26 May 2017, section 6) that the decision under appeal be set aside and that a European patent be granted on the basis of the main request, filed 17 June 2009, or on the basis of the first auxiliary request, filed 4 October 2011, or on the basis of the second auxiliary request, filed 26 May 2017.

The appellant further requested the referral of two questions to the Enlarged Board of Appeal.
The further application documents are: description pages 3-68 as originally filed, pages 1, 2, 2a filed on 8 February 2006; drawing sheets 1-5, 7-13 as originally filed, and sheet 6 filed on 28 October 2011.

VII. Claim 1 of the main request reads as follows:

"1. A system for allowing a client application (212, 260) to access a service at a server (214, 215), wherein the server includes a server engine (201) that provides access to the service via an interface (216, 217), the system characterised by:

a set of implementations (221, 222, 225, 226) of an interface (216,217) that is registered with the server engine to provide various functions of the service, wherein the implementation comprises an array of pointers (270, 272, 274) that point to the various functions provided thereby, wherein said implementation is dynamically linked and loaded into the server engine, and wherein the interface is dynamically customized by the implementation and the implementation is loaded into the client application’s address space;

a realization function (262) that allows the client application to send a request (266) to realize an interface to the service; and

wherein upon receiving the request to realize the interface, the server engine is adapted to locate the implementation, retrieve the array of pointers associated with the implementation, copy the array of pointers to a proxy array, and return a pointer to the proxy array to the client application, for use by the client application in retrieving the implementation into local memory, and accessing the service."

VIII. Claim 1 of the first auxiliary request reads as follows:

"1. A system for allowing a client application (212, 260) to access a service at a server (214,215), wherein the server includes a server engine (201) that provides access to the service via an interface (216,217), the system characterised by:

a set of implementations (221, 222, 225, 226) of an interface (216,217) that is registered with the server engine to provide various functions of the service, wherein the implementation comprises an array of pointers (270, 272, 274) that, point to the various functions provided thereby, wherein said implementation comprises a plugin module that is dynamically linked and loaded into the server engine, wherein the implementation is loaded into the client application's address space, and wherein the interface is dynamically customized by the implementation by providing plugin application programming interfaces that act as extensions to the interfaces provided by the server engine;

a realization function (262) that allows the client application to send a request (266) to realize an interface to the service; and

wherein upon receiving the request to realize the interface, the server engine is adapted to locate the implementation, retrieve the array of pointers associated with the implementation, copy the array of pointers to a proxy array, and return a pointer to the proxy array to the client application, for use by the client application in retrieving the implementation into local memory, and accessing the service."

IX. Claim 1 of the second auxiliary request reads as follows:

"1. A system for allowing a client application (212, 260) to access a service at a server (214,215), wherein the server includes a server engine (201) that provides access to the service via an interface (216, 217), the system characterised by:

a first implementation (221) of an interface (216,217) that is registered with the server engine to provide various functions of the service, wherein the first implementation comprises a first array of pointers (270, 272, 274) that point to the various functions provided thereby, wherein said first implementation comprises a plugin module that is dynamically linked and loaded into the server engine, wherein the first implementation is loaded into the client application’s address space, and wherein the interface is dynamically customized by the first implementation by providing plugin application programming interfaces that act as extensions to the interfaces provided by the server engine;

a realization function (262) that allows the client application to send a request (266) to realize an interface to the service; and

a second implementation of the same interface, wherein said first implementation inherits, from the second implementation, a subset of functions, the second implementation comprising a second array of pointers (290) that point to the various functions provided thereby;

wherein upon receiving the request to realize the interface, the server engine is adapted to locate the first implementation, retrieve the first array of pointers associated with the first implementation, copy the first array of pointers to a proxy array, copy pointers for the subset of functions from the second array of pointers to the proxy array and return a pointer to the proxy array to the client application, for use by the client application in retrieving the first implementation into local memory, and accessing the service."

Reasons for the Decision

1. *Summary of the invention*

The application relates to a piece of software called a "plugin framework" (see original description paragraph [15], first sentence; abbreviated PIF, see [73], fourth sentence and [32], second sentence) on a single computer (e.g. an e-commerce server, [8]). In the PIF, a so-called "(application) server engine" program 201 allows dynamic customisation of software interfaces by extending the PIF through plugin software modules

([15], first sentence; figure 3: 201, 216, 217). In this context, "dynamic" means "at runtime" ([37], last sentence; original claims 3 and 12). The PIF contains two kinds of software components, the client components (called "client application (212,260)" in the claims, "caller 260" in figure 6, "Applications 206" in figure 2 or "client applications 206" in [32], sixth sentence) and the server components (called "server (214, 215)" in the claims, "Server (Provider) A/B 214/215" in figure 3). A client component invokes a service provided by a server component through an interface 216/217 ([36]; figure 3).

An interface is a data type such as, for example, a "struct" in the C programming language or an array where each element of the data structure is a function pointer (i.e. an address of a C function procedure) pointing to a plugin function implementing the service ([38], first sentence; [75], second sentence; [31], page 8, lines 15-17). The interface must be registered and "realized" in the PIF ([38], second sentence). After the registration of interface implementations (which are, for example, contained in a DLL file) in the framework registry ([69]), a client application component makes a request to the PIF to "realizing" the interface by calling the _e_pif_realize routine ([70], sentences 1-3; [73], fourth sentence; [74], fifth sentence and figure 6: call 266 of _e_pif_realize). This call results in the client receiving a pointer to the desired implementation, which it uses to "retrieve" the implementation (i.e. the DLL) into its local memory ([73], fifth sentence; figure 5: 256 "Load Implementation Code into Client Memory"; see also claim 1 of the current main request, lines 18-19). This "retrieving the implementation into local memory"

during the realization process is also designated as "loading it [meaning the implementation] into the caller's address space" ([38], fourth sentence; see also claim 1 of the main request, lines 11-12). After the realization, the client application can invoke the functions in the implementation DLL ([38], third sentence).

2. *Overview of the present decision*

2.1 The clarity objections in the first instance decision (18.1-18.2) are unjustified.

2.2 The board does not remit the case to the department of first instance (Article 111(1) EPC 1973).

2.3 Claim 1 of all requests lacks an inventive step (Article 56 EPC 1973).

2.4 The board does not refer either of the two questions submitted by the appellant to the Enlarged Board of Appeal (Article 112(1)(a) EPC 1973).

3. *Clarity*

3.1 Apparently, most of the clarity objections in the decision (18.1-18.2) stem from the fact that the decision assumes that the "client application" and the "server" of the claims interact over a network, i.e. that they run in a client-server (hardware) architecture.

3.2 For example, the decision (18.1) states that "more than just a pointer might be transferred to the client". However, as argued in the grounds of appeal (3.4), the

application does not mention any transferral (or transmission of data over a network), but relates to a client software component "asking" a service provided by another component (the server component; see [16]). The board could not find any occurrence of the words "transfer", "transmit" or "network" in the description, nor any network-connected computer in the figures, but only software entities.

3.3 In fact, the "client-server relationship" in [16] and [36] could better be designated as "caller-callee relationship" or "invocation relationship", see [36], second sentence:

> "A component 212 is considered a client when it invokes a service 220,224 provided by another component, and a server 214,215 when it provides a service being invoked by another component."

That this invocation relationship is happening at the same computer follows from the claimed loading of the implementation into the client's address space (see also [38], fourth sentence) and the client application retrieving the implementation into local memory (see also [73], fifth sentence). Otherwise, a download of the implementation from a server computer to a client computer would have to be disclosed in the description, which is not the case.

In addition, it is noted that this usage of the term "server" as a software component reminds the board of the terminology in the so-called "COM" framework from Microsoft, see for example https://de.wikipedia.org/w/index.php?title=Component_Object_Model&oldid=162697656:

"Unter einem COM-Server versteht man ein Laufzeitmodul (Dynamic Link Library) oder ein ausführbares Programm, das in einer COM-unterstützenden Programmiersprache erstellt wurde und COM-Komponenten anbietet und erstellen kann." (A COM-server is understood as a run-time module (dynamic link library) or an executable program which was created in a programming language supported by COM and is able to provide and create COM-components.)

3.4 As to the clarity objection in the decision, section 18.2, the board agrees with the grounds of appeal (4.2) that the description clearly shows that a Vtable (an array of function pointers) is not identical to the implementation, but merely an array or a "struct" of pointers to the implementing functions (see [38], figure 6, [74], [75]). According to [74], there are two Vtables, one (270 in figure 6; "plugins Vtbl 270" in [74], seventh sentence) which claim 1 of the main request (line 8) designates as being comprised in the implementation, and the other ("Proxy Vtable 276" in the corrected version of figure 6, filed on 26 October 2011; also "proxy Vtbl 276" in [74], seventh sentence) which claim 1 (lines 17-18) terms a "proxy array", created by copying the first one.

3.5 As to the last paragraph of section 18.2 on page 6, the objection that "is not clear from the claim where the proxy array is situated (client-side, server-side, or elsewhere)" is also based on the previously mentioned misunderstanding that the claimed system is a client-server (hardware) architecture.

3.6 Therefore, the clarity objections in the decision are unjustified (Article 84 EPC 1973). Consequently, the appeal is allowable in the sense that the reasons of the decision do not justify a refusal of the application. On the other hand, the examination of the application has to be continued, because in the first instance there was no decision on the patentability of the claimed subject-matter.

4. *No remittal to the department of first instance*

Article 111(1) EPC 1973 reads: "Following the examination as to the allowability of the appeal, the Board of Appeal shall decide on the appeal. The Board of Appeal may either exercise any power within the competence of the department which was responsible for the decision appealed or remit the case to that department for further prosecution." The appeal is allowable, because the examining division erred in its decision that certain claims of the application lacked clarity (see section 3 supra). Making use of its discretion, the board decides to exercise the power within the competence of the examining division to examine the subject matter of the claims submitted for a decision on appeal. Accordingly, the board will assess whether the subject-matter of claim 1 of each request involves an inventive step. It is true that proceedings before the EPO are designed such that issues may normally be decided by two instances, i.e. an administrative first-instance department and, upon judicial review, by the boards of appeal. However Article 111(1) EPC allows for an exception to this rule. Since the board is conscious of the negative opinion of the examining division on the presence of an inventive step (see "obiter dicta", sections 21 and 22

of the decision under appeal), it would serve no purpose to remit the case. The board is aware that this opinion can only be derived from "obiter dicta" on inventive step and that a decision by the examining division on this subject would have carried greater weight. But, as the argument of the examining division is sufficiently elaborated and clear and the amendments before the board have not brought about substantial changes that would affect the assessment of the case, the board considers that the opinion of the examining division would not be substantially affected by arguments put forward in the appeal proceedings. Consequently, the board decides itself on inventive step.

5. *Inventiveness of claim 1 of the second auxiliary request*

5.1 In the following, only the second auxiliary request is analysed, since it is more concrete and narrower than the other requests. Therefore, the finding of lack of inventive step relating to the second auxiliary request also applies to the main request and the first auxiliary request.

5.2 Initially, the board noted in its communication sent in preparation for the oral proceedings that several features do not contribute to the presence of an inventive step, since they are merely present somewhere in the claim, but are not referred to elsewhere in the claim. Therefore the board concludes that they do not have any function in the claimed system and therefore do not contribute to inventive step. These features are:

- the "server (214, 215)" (line 2): in the following, only the "server engine" (included by the server) is used; nothing is needed from the remaining server (whatever this might be);
- the registration [of the first implementation] (line 6);
- a plugin module [comprised in the first implementation] (line 10): not used in the following;
- the dynamic linking and loading of the plugin module of the first implementation into the *server engine*: this first loading of the implementation (lines 9-10), which targets the server engine's address space, seems to be useless, since the client cannot call it from there by inter-process communication, and the same implementation is anyway loaded a second time, namely into the *client application's address space* (line 11) in which the client application can easily invoke it;
- providing plugin APIs that act as extensions to the interfaces provided by the server engine: not used in the following.

5.3 The *essence of the claimed invention* can be summarised as follows:

- a client application calls a "realization" function at a server engine in order to invoke the functions of a first implementation;
- in response to the realization request, the server engine locates the first implementation,
- copies a first array of pointers pointing to the implemented functions from the first implementation to a so-called "proxy" array,

- copies pointers from a second array of pointers to the proxy array for a subset of function inherited from a second implementation and
- returns a pointer to the proxy array to the client application;
- furthermore, the first implementation is loaded into the client application's address space and retrieved from the local memory of the client application process.

5.4 Document D3, which has already been selected as closest prior art, in the decision's obiter dictum (21.1) discloses that:

- a client application calls a "realization" function at a server engine in order to invoke the functions of an implementation: see page 56, lines 1-3: "A client instantiates a component instance by calling the COM CoCreateInstance() function and specifying the desired CLSID and IID, ..." - Thus, in D3 the "realization" function is called CoCreateInstance() function, the server engine is called COM runtime system;
- as a response to the realization request, the server engine locates the implementation: implicit, since the server engine/COM runtime system of D3 later on loads the implementation into the client component's address space;
- returns a pointer to the table of pointers pointing to the implementing functions: see page 56, lines 3-5: "... and the COM runtime system returns a pointer to the desired interface." - The interface of D3 is a table of function pointers/Vtable, see page 55, right column, lines 6-9: "It [= COM] implements component interfaces as a table of function pointers, which are called vtables ...";

- the implementation is loaded into the client application's address space/local memory of the client application process: see page 56, fourth paragraph: "For in-process instances, the component implementation is usually held in a dynamically linked library (DLL) and is loaded directly into the process's address space. The vtable then points directly to the component implementation." - The execution context used in the claim is called "in-process" in D3; see page 56, third paragraph: "Under COM, components can be instantiated in three different execution contexts: directly in the application's process (in-process), ..."

5.5 Thus, the claim differs from D3 in that it uses an array as the data structure for representing the table of function pointers and in that it copies the array from the implementation to a so-called "proxy array". Furthermore, function pointers from a second implementation are also copied to the proxy array in order to let the first implementation inherit functions from the second implementation.

5.6 The board cannot identify a technical effect emerging from these differences which goes beyond the usual effects of using arrays, function pointers, copying and inheritance, well-known before the priority date of this application.

5.7 The appellant argued that the invention avoided having to re-program the functions of the second implementation in the first implementation. This allowed the reuse of program code and reduced development time.

5.8 However, this merely eases and accelerates the programming, which is a non-technical activity. Therefore this cannot contribute to the technical character of the invention, and consequently also not to the presence of an inventive step (see also decision T 423/11 of the same board, sections 3.9, 3.11 and 3.12).

5.9 The appellant further argued that the reuse of code reduced the memory consumption.

5.10 This did not convince the board either, since a memory reduction would only occur compared to a hypothetical programming situation in which the programmer chose to first write the code of the functions for one implementation and then again for the other implementation. However, in the system of D3 the programmer would directly write the code of the functions he wanted to use, and this only once.

5.11 Furthermore, if a situation arose in which the programmer - for technical reasons - needed the same function in two different implementations, he would not hesitate to access an existing function implementation by a function pointer, since D3 already uses Vtables and function pointers.

5.12 Therefore, the subject-matter of claim 1 of the second auxiliary request is not inventive (Article 56 EPC 1973).

5.13 As stated above at section 5.1, the same arguments apply to claim 1 of the main request and the first auxiliary request.

6. *No referral to the Enlarged Board of Appeal*

Referring to the board's statement made in section 5.6, the appellant drew attention to section G VII 3.1 of the Guidelines that reads: "an assertion that something is common general knowledge need only be backed by documentary evidence (for example, a textbook) if this is contested" and submits that this passage is binding on the boards of appeal when exercising the power of the examining division. In the case at hand, the board denied that it was obliged to provide written evidence and explained during the oral proceedings that its members knew from their work on previous cases that the features listed supra were known to the person skilled in the art. It would therefore have been for the appellant to show an error in the board's appreciation, e.g. that one of the features at issue was in fact introduced into the art after the priority date of the application under appeal. The appellant sees in this discrepancy a point of law of fundamental importance for which he seeks clarification and requests the referral of one or both of the following questions to the Enlarged Board of Appeal.

6.1 The first question

> "To what extent is a Board of Appeal, when exercising power within the competence of the first instance department which was responsible for the decision appealed under Article 111(1) EPC, subject to the same constraints on that power as the first instance department, such as the duty to follow the Guidelines?"

The board judges that an answer to this question is not required. In order to ensure uniform application of the law, or if a point of law of fundamental importance arises, the board shall, during proceedings on a case and either of its own motion or following a request from a party to the appeal, refer any question to the Enlarged Board of Appeal if it considers that a decision is required for the above purposes (Article 112(1)(a) EPC). The Guidelines for Examination at the European Patent Office (Version November 2016 - further referred to as the Guidelines) are not part of the European Patent Convention (see Article 164(1) EPC 1973) and therefore cannot be binding upon the members of the boards of appeal (see Article 23(3) EPC 1973). This is acknowledged in the Guidelines on page 5, paragraph 1, that reads:

> "It should be noted also that the Guidelines do not constitute legal provisions. For the ultimate authority on practice in the EPO, it is necessary to refer firstly to the European Patent Convention itself including the Implementing Regulations, the Protocol on the Interpretation of Article 69 EPC, the Protocol on Centralisation, the Protocol on Recognition, the Protocol on Privileges and Immunities and the Rules relating to Fees, and secondly to the interpretation put upon the EPC by the Boards of Appeal and the Enlarged Board of Appeal."

6.2 The second question

> "To what extent is a board of Appeal, when exercising power within the competence of the first instance department which was responsible for the

> decision appealed under Article 111(1) EPC, entitled to rely upon an assertion of common general knowledge, such as for interpreting a prior art citation or assessing a technical effect, without providing documentary evidence of such common knowledge, even though the assertion is contested and there is, in effect, no possibility of appeal against such assertion."

The board refuses this request, as the question can be answered by reference to the EPC without doubt (see Case Law of the Boards of Appeal, 8th edition, IV., F., 2.3.7).

Firstly, the board refers to its answer to the first question: there is no obligation on the part of a board of appeal to follow the Guidelines. Secondly, there is also no general obligation on a board to provide documentary evidence for the existence of a piece of common general knowledge. In proceedings before the EPO, a board of appeal has to respect the right to be heard and to give reasons for its decision. In cases where a board refers to common general knowledge as state of the art, it is not obliged to provide documents in every conceivable case. Doing so may be the most straightforward way to give convincing reasons why, for example, a certain feature was known in the art at the priority date of an application. But it is not the only way. It is also possible for a board to state what it deems to be known, and possibly where it is known from, in a way that puts the appellant in a position to try to convince the board that its findings are erroneous. Proceeding that way respects the party's right to be heard (see for example T 458/07, section 7 and R 20/11, sections 3.1 and 4, last sentence). In

analogy to the situation in which documentary state of the art is presented, the appellant has the opportunity to comment on the board's view and provide arguments as to why the facts the board relies on are held to be wrong. A mere denial that such commonly known prior art existed is not sufficient. In its letter of 26 May 2017, the appellant complained that it was not in a position to prove a negative, i.e. what is not known to the skilled person. But that is not what the appellant is supposed to do. He could, for example, have provided evidence of when the contentious array data structure feature was introduced into the art, which the appellant however chose not to do.

Order

For these reasons it is decided that:

The request for referral of questions to the Enlarged Board of Appeal is refused.

The appeal is dismissed.

The Registrar: The Chairman:

D. Magliano W. Sekretaruk

Decision electronically authenticated