Internal distribution code:

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

Datasheet for the decision of 17 October 2014

Case Number: T 0039/11 - 3.5.01

Application Number: 03723394.7

Publication Number: 1507211

IPC: G06F12/14, G06F9/44, H04B7/26, H04M1/00, H04M1/725

Language of the proceedings: EN

Title of invention:
COMMUNICATION DEVICE

Applicant:
NTT DoCoMo, Inc.

Headword:
Data access control/NTT DOCOMO

Relevant legal provisions:
EPC 1973 Art. 56

Keyword:
Inventive step - (no)

Decisions cited:
T 0641/00

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 0039/11 - 3.5.01

D E C I S I O N
of Technical Board of Appeal 3.5.01
of 17 October 2014

Appellant: (Applicant)	NTT DoCoMo, Inc. 11-1, Nagatacho 2-chome, Chiyoda-ku Tokyo 100-6150 (JP)
Representative:	Hoffmann Eitle Patent- und Rechtsanwälte PartmbB Arabellastraße 30 81925 München (DE)
Decision under appeal:	**Decision of the Examining Division of the European Patent Office posted on 15 July 2010 refusing European patent application No. 03723394.7 pursuant to Article 97(2) EPC.**

Composition of the Board:

Chairman	W. Chandler
Members:	P. Scriven
	S. Fernández de Córdoba

Summary of Facts and Submissions

I. This appeal is against the decision of the Examining Division to refuse the European patent application No. 03723394.7.

II. The Examining Division decided on the basis of a main request and four auxiliary requests. The fourth auxiliary request was refused for lack of inventive step in view of *D2: Gong L: "Java security architecture (JDK 1.2), version 1.0", 6 December 1998, retrieved from the Internet on 9 July 2001.*

III. In the statement setting out the grounds of appeal, the appellant requested the grant of a patent based on a main request, corresponding to the refused fourth auxiliary request before the Examining Division, or else on first or second auxiliary requests, filed with the statement of grounds. The appellant requested oral proceedings be held "in the event that refusal of the application is contemplated."

IV. In a communication accompanying a summons to oral proceedings, the Board set out its preliminary opinion that none of the requests appeared to involve an inventive step in view of a number of Java programming tools such as encapsulation and the access permissions disclosed in D2.

V. With a letter dated 17 September 2014, the appellant provided additional arguments in favour of inventive step, and filed an amended first auxiliary request. The first and second auxiliary request filed with the grounds of appeal were maintained and renumbered as the second and third auxiliary request, respectively.

VI. Oral proceedings before the Board took place on 17 October 2014 with the appellant present. The appellant requested that the decision under appeal be set aside and that a patent be granted on the basis of the main request corresponding to the fourth auxiliary request refused by the Examining Division, the first auxiliary request filed on 17 September 2014, or the second or third auxiliary request corresponding to the first and second auxiliary requests filed with the statement setting out the grounds of appeal.

VII. Claim 1 of the main request reads:

"A communication device comprising:

a receiving means for receiving a program in byte code;

a specifying means adapted to specify data to be used from among data stored in the communication device when a program received by said receiving means is executed;

a first generation means for generating, when the received program is to be executed, a perfect encapsulated object having a method which is for processing encapsulated data from an outside object, the object having the encapsulated data specified by said specifying means, and which is adapted to deny access to said encapsulated data by said executed program received by said receiving means, wherein the generation means is for generating a perfect encapsulated object or an imperfect encapsulated object depending on a type designation information associated with the specified data, said perfect encapsulated object not having a method for authorizing access to

encapsulated data by an executed program;

an access control means for restricting accessible resources, and prohibiting access to data specified by said specifying means from among data stored in the communication device."

VIII. The first auxiliary request adds to claim 1 of the main request the feature that the imperfect encapsulated object has at least one method for authorizing access to encapsulated data by said executed program.

IX. Claim 1 of the second auxiliary request differs from the main request by the addition of the feature

"means comprised in an application manager, for determining whether an object for processing said specified data is a perfect encapsulated object or an imperfect encapsulated object on the basis of type designation information associated to the specified data"

and by the slightly different definition of the first generation means, as being

"comprised in said application manager, for generating, when the received program is to be executed, a perfect encapsulated object or an imperfect encapsulated object on the basis of the determining of said means for determining, wherein said perfect encapsulated object is an object configured not to have a method for authorizing access to encapsulated data by an executed program and to have a method which is for processing encapsulated data from an outside object, the object having the encapsulated data specified by said specifying means and which is adapted to deny

access to said encapsulated data by said executed program received by said receiving means, and wherein said imperfect encapsulated object is an object configured to have at least one method for authorising access to encapsulated data by said executed program".

X. The third auxiliary request differs from the main request in that the communication device in claim 1 comprises an additional

"program determining means for determining whether the received program is a downloaded Java application or a native application",

and in that the specifying means is adapted to specify

"when the program determining means determines that the program is a downloaded Java application, data to be used from among data stored in the communication device when a program received by said receiving means is executed by analyzing the program content of the Java application".

XI. The appellant's arguments, in so far as they are relevant to the Board's decision, may be summarized as follows:

The prior art allowed either full or no access to data whereas the invention allowed something in between.

It was not known from the prior art to use "perfect encapsulated objects". In object-oriented programming, an object containing private data fields would typically include methods for accessing the data.

It was not known to use encapsulation for data

protection purposes. The motivation behind encapsulation in object-oriented programming was normally to protect code from accidental corruption, and not to protect data from malicious code. Using perfect encapsulated objects to prevent access to phone data would, therefore, be hindsight.

The "specifying means" in claim 1 specified the data to be used by analysing the program content of the downloaded application. The analysis was performed before the app was executed. The same was true for the generation of the encapsulated objects. This was not known from the prior art, and had the effect of avoiding delays at run-time.

XII. At the end of the oral proceedings, the Board announced its decision.

Reasons for the Decision

1. *The invention*

1.1 The invention concerns the protection of sensitive data, such as the user's address book and telephone numbers, stored in a mobile phone. It would be a security risk if this information were freely accessible to downloaded Java applications (apps) that could try to steal data.

1.2 The invention seeks to overcome this by providing the phone data in an encapsulated object (figure 5). In this way, downloaded apps may not access the data

directly, but may only interact with encapsulated data via the public methods provided in the object. The invention uses two types of encapsulated object: "perfect encapsulated objects" and "imperfect encapsulated objects" (see point 2.9 for details). The type of encapsulation depends on the type of data to be encapsulated. For example, the address book is particularly sensitive, and, therefore, is provided in a "perfect encapsulated object" (see application at page 11, lines 9-24).

2. *Main request - claim 1*

2.1 The Board takes a mobile phone capable of downloading an executing Java apps as the starting point for the invention. It is common ground that such devices were prior art at the date of the invention (see the application at page 1, lines 10-14)

2.2 Java is an object-oriented programming language. A fundamental concept in all object-oriented programming is encapsulation. The idea is to make data fields private within the class that declares them, and to provide access, if at all, via public methods.

2.3 Apart from encapsulation, Java makes use of "the sandbox model" (see for example D2, section 1.1. "The Original Sandbox Model"). The essence of this model is that native applications are trusted to have full access to system resources, such as files, while downloaded apps are not trusted and are only allowed to access a limited set of resources inside the "sandbox" (see also the application, page 1, line 25 to page 2, line 1).

2.4 In view of this prior art, the Board considers that the following features of claim 1 were known in combination:

A communication device comprising:

A "receiving means for receiving a program in byte code". Here, "byte code" refers to compiled code that runs on the Java Virtual Machine.

A "specifying means adapted to specify data to be used from among data stored in the communication device when a program received by said receiving means is executed". The Board considers the specification of data to be implicit in any program.

A "generation means for generating [an] encapsulated object having a method which is for processing encapsulated data from an outside object, the object having the encapsulated data specified by said specifying means". This is nothing more than an encapsulated object having some data from the phone and at least one method.

An "access control means for restricting accessible resources, and prohibiting access to data specified by said specifying means [...]". The Board considers that encapsulation is in itself an "access control means" that falls under the definition in claim 1. Access control to data is, furthermore, provided both by the operating system and by the Java sandbox model.

2.5 The appellant argued that the definition of the "specifying means" in claim 1 implied an analysis, before the app was run, of the Java app in order to identify the data that was going to be used.

Encapsulated objects were also generated before the app was run. This was different from the known Java execution environment, in which the objects were generated at run-time.

2.6 In the Board's view, however, the wording *"when a program is [...] executed"* rather implies that the specifying of the data and the generation of the objects take place at run-time, as in the prior art. In that context, the wording *"when ... is to be executed"* in the definition of the generation means does not imply encapsulation before run-time.

2.7 It follows that the prior art set out above discloses the whole subject-matter of claim 1, except for the generation of a "perfect encapsulated object" or an "imperfect encapsulated object", depending on the type of data.

2.8 Claim 1 defines a "perfect encapsulated object" as having "a method which is for processing encapsulated data from an outside object" but not "a method for authorizing access to encapsulated data by an executed program". The appellant explained that, while an imperfect encapsulated object has a method which returns the encapsulated data to the object that invoked the method (e.g. getBytes()), a perfect encapsulated object does not contain any such method. A perfect encapsulated object only contains methods which will do something with the encapsulating data at the invoking object's request, such as displaying it on the screen (e.g. drawString()), and so provide access in some broader sense. The Board considers the appellant's interpretation to be a reasonable one that falls under the terms of the claim, and that is supported by the application as filed (e.g. page 18, line 29 to page 19,

line 25; figures 6 and 7). Therefore, the Board accepts this interpretation for the purpose of assessing inventive step.

2.9 Thus, the distinguishing features of the invention are directed to providing more or less access to encapsulated data, depending on the type of data. This is essentially the Examining Division's reasoning in the decision under appeal.

2.10 In the Board's view, the aim of protecting sensitive information is not technical, and may legitimately appear in the formulation of the technical problem (T 641/00 - "Two identities/Comvik", OJ EPO 2003, 352). Therefore, the Board considers that the problem solved by the invention is controlling access to data stored in the communication device, in the light of the non-technical requirement of protecting sensitive data.

2.11 The skilled person, who is a Java programmer, would inevitably have had to consider how to define her objects/classes, including which forms of access to give to encapsulated data, i. e. which accessor methods to provide, depending on the desired functionality and degree of protection. If "get"-access to a particular type of data is not desired for security reasons, the Board's opinion is that it would be self-evident to exclude methods such as getByte().

2.12 The appellant argued that the purpose of using encapsulation in object-oriented programming was different from that achieved by the invention. According to the appellant, the motivation behind encapsulation was normally to protect code from accidental corruption, and not to protect data from

malicious code. In other words, it was not known to use encapsulation for data security purposes.

2.13 In the Board's view, however, the access control in claim 1 is not limited to any particular purpose or intention. Indeed, the difference between malicious code and unintentionally erroneous code lies entirely in the mind of the programmer writing the code. The Board considers that the general idea behind encapsulation is to protect the object's data components from access by unauthorized code defined outside the class, and that the skilled person would have considered it as a means to protect sensitive data stored in the mobile phone.

2.14 Thus, in the Board's judgement, the subject-matter of claim 1 does not involve an inventive step (Article 56 EPC 1973) in view of a communication device comprising the standard Java environment.

3. *First auxiliary request - claim 1*

3.1 Claim 1 provides a definition of the "imperfect encapsulated object" as "having at least one method for authorizing access to encapsulated data by said executed program". The Board understands this to mean that an imperfect encapsulated object has a accessor method such as getBytes().

3.2 The Board's analysis of the main request takes account of just such a definition. As a consequence, claim 1 of the first auxiliary request lacks inventive step (Article 56 EPC 1973) for the same reasons as claim 1 of the main request.

4. *Second auxiliary request - claim 1*

4.1 Claim 1 of the second auxiliary request differs from the first auxiliary request essentially only in that the generation means is comprised in an "application manager".

4.2 The applicant argued that the "application manager" referred to a Java application manager (JAM) which was run when a program to be executed was designated from a program list (see the application, page 22, lines 6-9). This feature was said to further clarify that the encapsulated objects were generated before the downloaded application was executed.

4.3 The Board sees the amendment as giving a name to what was necessarily already there. Therefore, the second auxiliary request does not provide any features over the previous requests, which could establish an inventive step (Article 56 EPC 1973).

5. *Third auxiliary request - claim 1*

5.1 Claim 1 according to the third auxiliary request makes a distinction between "downloaded Java applications" and "native applications". The encapsulated objects are generated for data to be used by downloaded apps.

5.2 However, the Java sandbox model is based on the same distinction between native applications (local code) and downloaded applications (remote code). Therefore, this feature of the third auxiliary request does not add anything new.

5.3 Claim 1 also defines that the specifying means specify data "by analyzing the program content of the Java application". The appellant explained, in the oral proceedings, that this was done by looking at the function calls within the program, before the program was executed. The appellant argued that this pre-analysis had the effect of avoiding delays at run-time.

5.4 The application does not contain any details as to how the analysis of the program content is to be carried out, so the Board assumes that this was well known. Furthermore, the effect of this feature is not a reduction in complexity; the processing is merely offset to an earlier stage, which is just why compiled programs have the same advantage over interpreted ones. In the case of Java byte code, the Java virtual machine checks type constraints before issuing instructions to the processor, thus generating run-time objects. It is, therefore, the Board's view that pre-analysis is, at best, a standard alternative.

5.5 In conclusion, claim 1 of the third auxiliary request does not contain any feature that establishes an inventive step over the known communication device providing a standard Java environment. The third auxiliary request is therefore not allowable for lack of inventive step (Article 56 EPC 1973).

6. Since there is no allowable request, the appeal must be dismissed.

Order

For these reasons it is decided that:

The appeal is dismissed.

The Registrar: The Chairman:

T. Buschek W. Chandler

Decision electronically authenticated