Internal distribution code:

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

Datasheet for the decision of 27 July 2023

Case Number: T 2263/19 - 3.5.06

Application Number: 10167783.9

Publication Number: 2287737

IPC: G06F9/46

Language of the proceedings: EN

Title of invention:
Data processor and data processing method

Applicant:
Clarion Co., Ltd.

Headword:
Data processor and data processing method/CLARION

Relevant legal provisions:
EPC Art. 56
EPC R. 103(1)(a)

Keyword:
Inventive step - (no)
Substantial procedural violation - (no)
Reimbursement of appeal fee - (no)

Decisions cited:
T 2235/12

Catchword:

Europäisches Patentamt
European Patent Office
Office européen des brevets

Beschwerdekammern

Boards of Appeal

Chambres de recours

Boards of Appeal of the
European Patent Office
Richard-Reitzner-Allee 8
85540 Haar
GERMANY
Tel. +49 (0)89 2399-0
Fax +49 (0)89 2399-4465

Case Number: T 2263/19 - 3.5.06

D E C I S I O N
of Technical Board of Appeal 3.5.06
of 27 July 2023

Appellant: (Applicant)	Clarion Co., Ltd. 5-35-2, Hakusan Bunkyo-ku Tokyo 112-8608 (JP)
Representative:	MERH-IP Matias Erny Reichl Hoffmann Patentanwälte PartG mbB Paul-Heyse-Straße 29 80336 München (DE)
Decision under appeal:	**Decision of the Examining Division of the European Patent Office posted on 20 February 2019 refusing European patent application No. 10167783.9 pursuant to Article 97(2) EPC.**

Composition of the Board:

Chairman	M. Müller
Members:	A. Teale
	K. Kerber-Zubrzycka

Summary of Facts and Submissions

I. This is an appeal against the decision, dispatched with reasons on 20 February 2019, to refuse European patent application No. 10 167 783.9 on the basis that the claims according to a main and two auxiliary requests did not comply with Articles 123(2) (added subject-matter) and 84 EPC (clarity).

II. In examination proceedings the following documents, amongst others, were cited:

D1: US 6 675 285 B1

D4: David K.Y. Yau, Simon S. Lam, "Adaptive rate-controlled scheduling for multimedia applications", IEEE/ACM Transactions on Networking, IEEE/ACM New York, US, vol. 5, no. 4, 1 August 1997, pages 475 to 488, XP058355586, ISSN: 1063-6692, DOI: 10.1109/90.649461.

D5: Tong Li et al., "Efficient and scalable multiprocessor fair scheduling using distributed weighted round-robin", ACM Sigplan Notices, ACM, 2 Penn Plaza, Suite 701, New York, NY 10121-0701, US, vol. 44, no. 4, 14 February 2009, pages 65 to 74, XP058320941, ISSN: 0362-1349, DOI: 10.1145/1594835.1504188.

III. A notice of appeal and the appeal fee were received on 26 April 2019, the appellant requesting that the decision be set aside and making an auxiliary request for oral proceedings.

IV. With a statement of grounds of appeal, received on 1 July 2019, the appellant submitted amended claims according to new auxiliary requests 1 to 3. The appellant requested that a patent be granted on the basis of the main request on file and said new auxiliary requests. The appellant also requested reimbursement of the appeal fee, alleging a substantial procedural violation, Rule 103(1)(a) EPC.

V. In an annex to a summons to oral proceedings the board set out its provisional opinion on the appeal, as follows. The board had doubts regarding the clarity of the claims of all requests, Article 84 EPC. The amendments to the claims seemed to satisfy Article 123(2) EPC. The board had doubts regarding the inventive step, Article 56 EPC, of the claimed subject-matter of all requests in view of the combination of D1, D4 and D5. Regarding the request for reimbursement of the appeal fee, the board had doubts whether a substantial procedural violation occurred, reimbursement being anyway dependent on the appeal being deemed allowable, Rule 103(1)(a) EPC.

VI. In a letter received on 26 June 2023 the appellant withdrew the request for oral proceedings and requested a decision "on the state of the file". The appellant made no substantive comments on the case. The Board cancelled oral proceedings.

VII. The application is being considered in the following form:

Description (all requests):
pages 1 and 3 to 25, as originally filed, and
pages 2 and 2a, received on 25 August 2016.

Claims:
Main request: 1 to 4, received on 2 November 2018.
Auxiliary request 1: 1 to 4, received with the grounds of appeal.
Auxiliary request 2: 1 to 2, received with the grounds of appeal.
Auxiliary request 3: 1 to 2, received with the grounds of appeal.

Drawings (all requests):
Pages 1/3 to 3/3, as originally filed.

VIII. Claim 1 according to the main request reads as follows:

"1. A data processor, comprising: a CPU (101); a plurality of peripheral devices (107; 108; 109; 110); and a memory (102) comprising: a plurality of device control threads (112; 113; 115; 117) for controlling operation of respective peripheral devices; a control thread (111) including a plurality of processing parts associated with respective device control threads; a table (118) storing configurations of a plurality of applications including application priority of respective applications, wherein each application includes an operation sequence of processing parts that constitute an application and a dependence relationship between the processing parts of the application, wherein the processing parts are prioritized such that processing part near end [sic] of the operation sequence is performed with higher priority than processing part near beginning of the operation sequence; the control thread is configured to perform the steps of: executing two or more of the applications sharing at least one peripheral device in parallel, wherein when an overlap between operation timings of respective processing parts of the two or more of the

applications corresponding to a same peripheral device occurs, processing part with higher priority is executed before respective processing parts of other applications, wherein if priorities associated with the respective processing parts are the same, application priority of application associated with processing part that last executed is lowered such that processing part of the other application with higher priority is executed, and if each of the respective processing parts has no dependence relationship based on the configuration stored in the table, no priority is given to the respective processing parts such that the respective processing parts are executed in alternate manner."

IX. In claim 1 of the first auxiliary request, compared to that of the main request, the following feature has been added

[a control thread] "for controlling parallel execution of device control threads and CPU processing threads"

and the following feature has been deleted

"wherein when an overlap between operation timings of respective processing parts of the two or more of the applications corresponding to a same peripheral device occurs, processing part with higher priority is executed before respective processing parts of other applications, wherein".

X. In claim 1 of the second auxiliary request, compared to that of the previous request, the following features have been added at the end:

"wherein the plurality of the applications are defined by the table (118), the control thread (ill) is configured to instruct execution of sequences of processing parts defining a target application, according to the application priority provided to each of the applications, and to lower the application priority of the target application versus other applications, when control of the target application is returned to the control thread (111), and among the applications the application priorities of which are equal, the application priority of an application corresponding to the processing part instructed last to be executed is lowered."

XI. In claim 1 of the third auxiliary request, compared to that of the previous request, the following features have been added:

[a control thread] "for controlling parallel execution of device control threads and CPU processing threads through processing parts corresponding to the respective threads;",

[the control thread] "performs processing of processing parts corresponding to the device control threads (112,113, 115,117) and the CPU processing threads (114,116) for each of which the output data exists according to a priority such that a processing part near end of an operation sequence of processing parts that constitute the application is performed with higher priority than a processing part near the beginning of the operation sequence" and

"wherein with the use of the application management table (118), the priority of the application is retained, among applications the priorities of which

are equal, the priority of an application corresponding to a thread instructed last to be executed is lowered, and execution of any thread of the device control threads (112,113,115,117) and the CPU process threads (114,116) that correspond to the processing parts constructing another application whose performance interval is open is instructed with a higher priority."

Reasons for the Decision

1. Admissibility of the appeal

In view of the facts set out at points I, III and IV above, the appeal fulfills the admissibility requirements under the EPC and is consequently admissible.

2. Summary of the invention

2.1 The application relates to a data processor, in particular to a car navigation system capable of not only navigating and displaying maps but also simultaneously running multimedia applications; see page 1, lines 6 to 16. The question arises of how to share the plurality of system peripheral devices between a plurality of application programs so as to maximise response speeds and data throughput rates; see page 1, lines 17 to 21.

2.2 The application solves this problem by providing a data processor (see figure 1) having an application management table (118) stored in system memory (102). The table stores the configuration, including the priority, of each of a plurality of applications (143-149), each

application comprising a sequence of processing parts, each processing part being allocated a priority.

2.3 By changing the relative priorities of processing parts, a processing part near the end of a sequence can be prioritized over a processing part at the beginning. In the case of input/output devices, the priority of the output device thread is made higher than that of the input device thread, thereby reducing memory occupancy and latency whilst increasing the throughput; see page 3, lines 1 to 11.

2.4 The application addresses two problems; see page 3, lines 12 to 21. The first relates to changing the sequence in which peripheral devices are used. The second problem relates to high priority threads having to wait for low priority ones to complete.

2.5 Figure 1 illustrates a data processor comprising a CPU (101) linked by a bus to a memory (102), a DVD/CD (107), a hard disk (HDD 109) and a display (110); see page 11, line 14, to page 13, line 3. According to page 12, lines 3 to 6, the CPU runs an application by executing a sequence of processing parts using a plurality of threads; see page 12, lines 4 to 6.

2.6 Regarding the threads, each peripheral device has a corresponding "device control" thread (112,113,115,117) and there are also two "CPU processing" threads (114,116) relating to computing activities; see bottom of figure 1; 112-117 and the paragraph bridging pages 12 and 13. A "control thread" (111) controls the parallel execution of the device control threads and CPU processing threads; see figure 1, top of memory structure. This control is achieved by making the priority of the control thread higher than those of the

other threads; see page 12, lines 15 to 16, and page 21, lines 6 to 11. According to page 12, lines 12 to 13, "... each processing part is included in the control thread 111", figure 1 showing the following "processing parts": "DVD read", "decryption", decompression", "HDD read", "HDD write", "map display data origination" and "display".

2.7 Figures 2 and 4 show examples of application management tables (118), figure 4 showing the applications having different priorities. The tables each list a sequence of processing parts (143-147), carried out by the device control threads (112-115, 117) and CPU processing threads (114,116), for each application (201, 202), these being "map updating" and "route guidance". For each processing part the processing part feeding it, termed the "preprocessing part", is indicated, understood by the board to imply a data dependency; the following operation can only start once the output data from the preprocessing operation is available.

2.8 The processing sequence for the map updating application (201) is as follows: DVD read (143), decryption (144), decompression (145) and HDD write (146); see page 14, lines 1 to 14. To ensure that processing occurs smoothly, the processing parts are operated with an increasing priority, in this case the priority climbing from DVD read (lowest priority) to HDD write (highest priority); see page 14, lines 15 to 20.

2.9 As shown in figure 1, the device control threads (112-115, 117) and CPU processing threads (114,116) each comprise an input buffer queue, an output buffer queue and a control queue (see page 15, line 5, to page 16, line 24). The application management table contains an output buffer flag (118) for each processing part

which indicates (flag="existence") that the processing part has completed, so that the output buffer contains output data; see the paragraph bridging pages 14 and 15. The control part (150) executes the device control threads (112-115,117) and the CPU processing threads (114,116); see the paragraph bridging pages 16-17 and page 17, lines 17 to 20.

2.10 Figure 3 shows the sequence of processing parts in the peripheral devices by the map updating application (201), the sequence of processing parts, namely DVD read (143), decryption (144), decompression (145) and HDD write (146), being shown in figure 2; see page 17, line 22. At time T1 the CPU executes the control thread (111) which checks the buffer output flags of all processing parts to see if the corresponding output buffer contains data. There is none. However the DVD read part (143) requires no input data, no preprocessing part being defined for it in figure 2, and is therefore executed by the control part (150) first; see page 17, line 24, to page 18, line 6. The DVD read part (143) enqueues a DVD read instruction, indicating the buffer in which the data is to be stored, in the DVD control thread control queue (119). The DVD control thread (112) sets the DVD/CD drive to store data in the designated buffer and waits until the DVD/CD drive is ready. At time T2 the DVD/CD drive starts reading, shown in figure 3 as period 301, under the control of the DVD read part 143, and sets the flag of the DVD read part output buffer to indicate that there is output data in the DVD part output buffer (121), completing at time T3 and returning control to the control part (150); see page 18, lines 14 to 24. The control part (150) then recognises that the precondition for executing the second step (decryption) is now fulfilled since, according to figure 2, the output data from the DVD read

part (143) feeds the decryption part (144). The control part gives a higher priority to the decryption part and executes it in time period 302. At the same time the DVD/CD drive can be used in period 305 to execute another processing part, the DVD/CD drive (107) and the encryption circuit (108) operating in parallel; see page 19, line 15, to page 21, line 5. In this way, and with increasing priority, the control part runs the decompression part (145) in time period 303 and the HDD write part (146) in time period 304 to complete the steps of the map updating application (201).

2.11 The second embodiment concerns the map updating application (201) and the route guidance application (202) running in parallel and sharing peripheral devices; see page 23, lines 1 to 17. Operations having no data dependence are given no priority and are executed alternately. The third embodiment focuses on the fluctuating use of the hard disk drive, for instance when the route guidance application (202) runs with a higher priority in parallel with the map updating application (201); see figure 4 and page 23, line 19, to page 25, line 12.

2.12 The claims of the main request concern two or more applications sharing the same peripheral device, prioritizing processing parts at the end of a sequence over those at the beginning and, if operation timings of two processing parts overlap and the application priorities differ, processing the processing part with higher priority, reducing the priority of the application whose processing part was last executed and executing the processing part of the now higher priority application, if the application priorities are the same, and, if the processing parts have no data dependencies, prioritizing neither and executing them alternately. Auxiliary request 1 is limited to the control thread (111) con-

trolling parallel execution of device control threads and CPU processing threads. Auxiliary request 2 is directed to lowering the priority of an application when control of that application is returned to the control thread. Claim 1 according to auxiliary request 3 is based on that as originally filed with the additional features of reducing the priority of the application whose processing part was last executed and then executing the processing part of the now higher priority application, if the application priorities are the same, and executing an application whose "performance interval is open" with a higher priority.

3. The board's understanding of the invention

3.1 In the light of figure 2, in particular, the board understands the control thread (150) to execute device control threads (112-115,117), relating to corresponding peripheral devices, and CPU processing threads, relating to computing steps. Each peripheral device and each of the two computing activities has a dedicated permanent thread comprising input and output queues (112,113,115,117).

3.2 Once a device control thread or a CPU processing thread terminates, control returns to the control thread. The control thread executes the sequence of processing parts required by the applications (201,202), stored in the application management table (118).

3.3 Each application is understood to have a default priority which can be "no priority"; see the table in figure 4, second column. The relative priorities are used to decide what to execute first. During execution the priority of individual processing parts can be changed from the default value for the application

according to the rules set out in claim 1. This sometimes changes the order in which processing parts are executed in the sense that, although data dependencies within application sequences (see figures 2 and 4) dictate a certain order, the order in which processing parts of different application sequences are processed can be changed.

3.4 The original claims set out the combination of the following three rules:

3.4.1 Rule 1: "... the control thread (111) ... performs with a higher priority from the processing part near termination of the processing of the application defined by the application management table (118) among the plurality of processing parts ..." This is set out in original independent claims 1 and 4.

3.4.2 Rule 2: "among applications the priorities of which are equal, the priority of an application corresponding to a thread instructed last to be executed is lowered". This is set out in original claim 3, which is dependent on claim 1, and occurs in the third embodiment; see page 23, line 19, to page 25, line 12.

3.4.3 Rule 3: "if each of the respective processing parts has no dependence relationship based on the configuration stored in the table, no priority is given to the respective processing parts such that the respective processing parts are executed in alternate manner." This is set out in original claim 8 which is dependent on claim 7, itself dependent on claim 4. This also occurs in the second embodiment; see page 23, lines 1 to 17.

3.5 According to page 12, lines 12 to 13, "... each processing part is included in the control thread 111."

The board understands this to mean that the control thread (111), acting as the control part (150), allocates processing parts to the input queues of the device control and CPU processing threads for execution.

3.6 Original claims 2 and 3 refer to an application whose "performance interval" is open. This expression, which is used in claim 1 as originally filed and that of auxiliary request 3, is also used in the following original paragraphs [19-23] which refer repeatedly to control returning to the control thread lowering the priority of the application that has just completed and initiating with a higher priority execution of any thread of the device control threads and the CPU processing threads corresponding to the processing parts that construct another application whose performance interval is open. Consequently the board understands an application whose "performance interval" is open to mean an application which is being executed but has not yet finished.

4. Clarity, Article 84 EPC

4.1 According to the reasons for the decision, the claims of the main request were unclear. The terms "thread" and "processing part" had to be treated as synonyms in order to interpret them meaningfully. This interpretation was supported by page 12, lines 2 to 6, of the original description. However this interpretation was contradicted by the present claims which set out "a common thread including a plurality of processing parts associated with respective device control threads". The nature and functioning of the processing parts was consequently unclear, as a thread was understood to be an atomic (i.e. indivisible) unit of execution which

could therefore not consist of entities which were separately prioritized and scheduled. Moreover, according to the claims, scheduling meant performing processing parts. In some cases, the processing parts had associated priorities which were used to schedule them. It was unclear how the scheduling system could combine the scheduling of processing parts having an associated priority with those without.

4.2 The board does not agree with this clarity objection. As set out above, the board does not understand a "thread" and a "processing part" to be the same. The input queues of the threads comprise a queue of processing parts waiting to be executed under the control of the control thread. The feature in claim 1 that the processing parts are included in the common thread is understood by the board to mean that the control thread (111), acting as the control part (150), has to "know" - i.e. "include" references to - the device control threads and the CPU threads. However, the skilled person would understand that the claims, when referring to "processing parts" being executed, refer to the individual tasks that eventually are, via the mentioned queues, executed via the individual threads.

4.3 The board further understands each application to have a default priority; see the table in figure 4, second column. However during execution the priority of individual processing parts can be changed from the default value for the application according to the rules set out in claim 1. For processing parts in different application sequences, which thus have no data dependencies, this can change the order in which the processing parts are executed.

4.4 Despite the further clarity issues raised by the board itself in the annex to the summons, the board finds that the claims of all requests are sufficiently clear for the purposes of assessing inventive step.

5. The prior art on file

5.1 Document D1

5.1.1 According to its abstract, D1 relates to eliminating "memory contention" in a multi-threaded computation engine for video graphics processing; see column 1, lines 7 to 9. Memory contention is avoided by each "data flow" having its own memory; see column 7, lines 22 to 24. As shown in figure 1, an "arbitration module" prioritizes instructions from several threads for execution by a computation engine accessing a memory.

5.1.2 Operations at the end of a sequence are prioritised to ensure efficient use of pipeline resources; see column 3, line 52, to column 4, line 3. The board regards this as "rule 1", set out above. The other two claimed rules are not known from the prioritization scheme shown in figure 7; see column 31, line 24, to column 32, line 43.

5.2 Document D4

According to its introduction, D4 concerns thread scheduling parallel multimedia applications in a CPU so as to meet certain Quality of Service (QoS) goals vis-à-vis the user. Page 1, right column, 1ines 30 to 32, discloses lowering the priority of a thread which has just been executed. The board regards this as "rule 2",

set out above. The other two claimed rules are not mentioned.

5.3 Document D5

According to its abstract, D5 relates to a "fair" (defined on page 66, section 2) scheduling algorithm for a multi-core processor which increases the priority of a thread that has not run for a while. The board finds that this is a generalised form of "rule 3", set out above; see page 65, right column, lines 8 to 11. The other two claimed rules are not mentioned.

6. Inventive step, Article 56 EPC

6.1 In a section entitled "Obiter dictum" (see points 22-23) the examining division made remarks referring to D1, D4 and D5 regarding inventive step, assuming that the expression "processing parts" in the claims meant "threads". The feature that, when multiple threads were ready to execute, the thread with the highest priority was executed, was seen as a usual scheduling technique. Reducing the priority of a thread that had just been executed was known from D4; see page 475, right column, lines 31 to 32. This also seemed to be a matter of fair scheduling; see D5, page 65, right column, lines 8 to 11. Prioritising threads nearing completion was known from D1; see column 3, line 52, to column 4, line 3. It could also be regarded as an example of the "earliest deadline first" scheduling algorithm; see D4, page 478, left column, lines 10 to 17.

6.2 The board finds that the engine known from D1 comprises a data processor including a CPU and peripheral devices. As set out above, it implements a thread scheduling method according to "rule 1". The skilled

person starting from D1 and solving the problem of providing an alternative scheduling scheme, an obvious problem starting from D1, would have considered adding "rule 2" from D4 and "rule 3" from D5 as a matter of usual design, the application giving no indication of the three rules together having an unexpected, synergistic effect.

6.3 Hence the board finds that the subject-matter of claim 1 of all requests does not involve an inventive step in the light of D1, D4 and D5.

7. The request for reimbursement of the appeal fee, Rule 103(1)(a) EPC

7.1 As one of the conditions for reimbursement of the appeal fee under Rule 103(1)(a) EPC is that the appeal be deemed allowable, which it is not, this request must be refused. Nevertheless the board comments as follows on whether a substantial procedural violation occurred.

7.2 The summons to oral proceedings of 16 May 2018 drew attention to documents D1 to D3 and raised a novelty objection based on D1. Thereupon the applicant filed a new main and two auxiliary requests. In a "brief communication" (to use the expression on EPO form 2009A) dated 3 December 2018, sixteen calendar days and eleven working days before the scheduled date of the oral proceedings of Wednesday 19 December 2018, the examining division raised further objections, namely added subject-matter, Article 123(2) EPC, and a lack of clarity of the expression "processing part", Article 84 EPC. These objections are the only reasons given in the decision regarding the main request. The division also cited documents D4 and D5 for the first time and made preliminary remarks based on D1, D4 and D5 as to

inventive step. These remarks do not form part of the reasons for the decision, but are included in the "obiter dictum" section at the end of it. The applicant replied in a telefax received on 12 December 2018, requesting a postponement of the oral proceedings, citing decision T 2235/12, to give the applicant's representative sufficient time to react to the new situation and to consult the applicant, a Japanese company. The examining division stated in a telefax dated 14 December 2018 that the oral proceedings would take place as planned. The applicant replied in a telefax dated 18 December 2018, the day before the oral proceedings, requesting a decision according to the state of the file and stating that it would not attend the oral proceedings. According to the minutes of the oral proceedings, the applicant was absent, as announced in advance, and the examining division announced that the proceedings would be continued in writing. The written decision was subsequently issued on 20 February 2019.

7.3 The appellant has argued that it had reacted as soon as possible to the "brief communication" by requesting a postponement of the oral proceedings, so that it could react, for instance by submitting comments or amending requests, to the new objections and obtain instructions from its client. The examining division's refusal of this request constituted a substantial procedural violation justifying reimbursement of the appeal fee. The period of eleven working days from sending the "brief communication" until the oral proceedings neglected a delivery period. Moreover the objection under Article 84 EPC concerning the expression "processing part" was raised for the first time in the "brief communication" and could have been raised earlier, for instance in the summons. It was inequitable vis-à-vis the applicant, in

examination proceedings spread over seven years, to cite new prior art just a few days before the oral proceedings.

7.4 The board notes that in T 2235/12 it was found that not postponing an oral proceedings after two new documents had been cited and new objections raised by the examining division one working day before the oral proceedings constituted a denial of the right to be heard, Article 113(1) EPC, and a substantial procedural violation; see reasons 3 to 3.9. In that case, one of the new documents formed part of the sole line of reasoning in the decision as to why claim 1 lacked novelty.

7.5 In the present case however, D4 and D5, first cited in the "brief communication" do not appear in the reasons for the decision. Hence the situation in the present case differs from that in T 2235/12. Consequently the question of a substantial procedural violation depends on whether eleven working days were sufficient for the applicant to react and be heard, Article 113(1) EPC, regarding the new objections under Articles 84 and 123(2) EPC contained in the "Brief communication".

7.6 The board finds that no procedural violation, let alone a substantial one, occurred. The appellant's arguments mainly concern the new documents, but these are not relevant because they do not form part of the reasons for the decision. The applicant could also have expected that the amendments filed in response to the summons might give rise to new objections; objections under Articles 123(2) and 84 EPC are not unusual when amended claims are filed. In the board's view, applicants must even be prepared for these types of objection to be raised for the first time in the oral proceedings themselves.

Order

For these reasons it is decided that:

1. The appeal is dismissed.
2. The request for reimbursement of the appeal fee is refused.

The Registrar: The Chairman:

L. Stridde M. Müller

Decision electronically authenticated