Internal distribution code:

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

Datasheet for the decision of 15 July 2021

Case Number: T 1120/16 - 3.5.06

Application Number: 07837681.1

Publication Number: 2057539

IPC: G06F9/44

Language of the proceedings: EN

Title of invention:
SPECIFYING IMPLEMENTATIONS OF CODE FOR CODE GENERATION FROM A MODEL

Applicant:
The MathWorks, Inc.

Headword:
Alternative implementations/MATHWORKS

Relevant legal provisions:
EPC 1973 Art. 56, 84
RPBA 2020 Art. 13(2)
EPC Art. 123(2)

Keyword:
Claims - clarity - main and second auxiliary request (no)
Amendment after summons - taken into account - first auxiliary request (no) - taken into account - third and fourth auxiliary requests (yes)
Inventive step - third and fourth auxiliary requests (no)

Decisions cited:
T 1294/16

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 1120/16 - 3.5.06

D E C I S I O N
of Technical Board of Appeal 3.5.06
of 15 July 2021

Appellant: (Applicant)	The MathWorks, Inc. 3 Apple Hill Drive Natick, MA 01760 (US)
Representative:	Rupprecht, Kay Meissner Bolte Patentanwälte Rechtsanwälte Partnerschaft mbB Widenmayerstraße 47 80538 München (DE)
Decision under appeal:	**Decision of the Examining Division of the European Patent Office posted on 18 December 2015 refusing European patent application No. 07837681.1 pursuant to Article 97(2) EPC.**

Composition of the Board:

Chairman	M. Müller
Members:	S. Krischer
	A. Jimenez

Summary of Facts and Submissions

I. The appeal lies from the decision of the examining division, with reasons dispatched on 18 December 2015, to refuse the European patent application No. 07 837 681 for lack of inventive step, Article 56 EPC 1973, over the document

D1: US 2005/177816 A1,

lack of compliance with Article 123(2) EPC and lack of clarity, Article 84 EPC 1973.

II. The applicant appealed this decision on 29 February 2016 and paid the due appeal fee. A statement of grounds of appeal was received on 25 April 2016. The appellant requested that the decision be set aside and a patent be granted on the basis of sets of claims according to a main request or one of auxiliary requests 1-3, all as filed with the grounds of appeal.

III. In an annex to the summons to oral proceedings, the board gave its preliminary opinion that claim 1 of all requests did not comply with Article 84 EPC 1973 and lacked inventive step over D1, Article 56 EPC 1973. A potential objection under Article 83 EPC 1973 was also mentioned.

IV. In response to the summons, the appellant field amended sets of claims (1-8, 1-8, 1-5, and 1-5, respectively) according to auxiliary requests 1-4.

V. During the oral proceedings, the appellant filed a new set of claims 1-4 according to an amended first auxiliary request.

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

"A method for generating code from a model, comprising:

performing each of the following steps with a computing system:

receiving a specification of a model scope wherein a region of the model is specified that includes at least one of a plurality of elements;

receiving a specification of an element scope that specifies a type of element in the model;

receiving a selection of an element in the model that has the specified model scope and the specified element scope;

when the element is selected, displaying an interface that includes designations that represent available implementations of code for the selected element, the available implementations of code comprising a first implementation of code and a second implementation of code;

receiving a specification of an implementation scope, the scope being one of the selected element, elements in the element scope, and the model scope, the specification of an implementation scope causing the interface to display designations of available implementations of code for the implementation scope;

receiving a selection of a designation, the selection specifying:

- a designation for performance characteristics or requirements, or

- a designation for a type of code, or

- a designation for a constraint including complying with a coding standard,

selecting one of the first implementation or the second implementation based on the designation specified by the selection; and

generating code for the model, wherein the selected one of the first or second implementations of code are used to generate code representing elements corresponding to the implementation scope, wherein the generated code is a programming language or a hardware description language."

VII. Claim 1 of the first auxiliary request is identical with claim 1 of the main request, except for two clarifications in the paragraph on "receiving a specification of an implementation scope", namely "the implementation scope being one the the selected elements" and the interface displaying designations "of available implements of code for the specified implementation scope" (additions underlined), and in that the last two paragraphs read as follows:

"... selecting one of available implementations based on the designation specified by the selection; and

generating code for the model, wherein the selected one of the available implementations are used to generate code representing elements corresponding to the implementation scope, wherein the generated code is a programming language or a hardware description language,

whereby the method is adapted to allow a user to generate multiple combinations of implementations of codes for a plurality of selected implementations as defined above."

VIII. Claim 1 of the second auxiliary request is identical with claim 1 of the main request,except that the selection is said to specify

"...

- a designation for an implementation of code that minimizes power, or
- a designation for an implementation of code that maximizes speed, or
- a designation for an implementation of code that minimizes area, or
- a designation for an implementation that minimizes memory use; ..."

IX. Claim 1 of the third auxiliary request corresponds to claim 1 of the second auxiliary request, except that the steps relating to the interface read as follows:

"... when the element is selected, displaying an interface that includes designations that represent available implementations of code for the selected element,

receiving a specification of a scope, the scope being one of the selected element, elements in the element scope, and the model scope, the specification of an implementation scope causing the interface to display designations of available implementations for the scope, wherein the available implementations of code for the scope and the selected element vary depending on a target language specified by the user; ..."

and the final generating step reads as follows:

"... generating code for the model, wherein the generated code is a programming language or a hardware description language, wherein one of a plurality of implementations for each of the target languages is used depending on the selection of designation and the specified target language."

X. Claim 1 of the fourth auxiliary request corresponds to claim 1 of the third auxiliary request, except that as the first step to be performed the following is specified:

"... generate a model based on a user's input the model being represented as a graphical model or a textual model and having a plurality of elements; ..."

XI. At the end of the oral proceedings, the chairman announced the decision of the board.

Reasons for the Decision

The invention

1. The application relates to code generation from a graphical "block diagram model" (page 1, lines 12-15 and 19-22). More specifically, the application considers graphical modeling environments such as Simulink and is concerned with aiding the developer of such a "model" (i.e. of a graphical computer program) in generating "deployable software" or "descriptions of hardware system" from it (see page 1, lines 24-26, and page 6, lines 8-16).

1.1 The model is defined in terms of elements (graphical or textual objects; see page 7, line 31, and figure 1). Via a suitable user interface, portions of the model can be selected. A preferred (and claimed) manner of doing that is by selecting first a "region of a model" (a "model scope"), then elements of a specific type (an "element scope"), and finally an individual element of the selected type in the selected region

(see page 16, lines 13-28; page 17, lines 10-16, and page 19, lines 27-33).

1.2 After that, the user interface presents implementations for the selected element (or another "scope the user wants to implement"; see page 20, lines 5-18; see also figures 7B, items 762 to 765, and figure 7D, items 772 to 778). These implementations may be marked as "meet[ing] various performance or design requirements" such as "power reduction" or "memory size reduction" or as being adapted to a desired "target device" ("FPGA, PLC, ASIC, ASIP, etc.") or "language" ("C, C++, Java, Javascript, VHDL, Verilog, etc."; see page 7, lines 2-5 and 13-15; page 13, lines 6-16 and paragraph bridging pages 13 and 14; page 14, lines 13-19; page 20, lines 20-26).

1.3 The user can pick one (possibly at a fee), use it unchanged or customize it, or decide to rather develop an own implementation from scratch (see e.g. page 22, paragraph 2, to page 23, paragraph 1, and page 24, lines 9-18; see also figure 8, items 820 and 830, and figure 10, item 1010). Eventually, a "code building tool" generates source code and compiles it so as to build an executable code (paragraph bridging pages 8 and 9).

1.4 In its submission of 15 June 2021, the appellant confirmed the board's paraphrase of the invention (see point 11).

The prior art

2. D1 discloses a system which enables the user to generate, from a graphical program and its "nodes" (i.e. "elements"; cf. paragraphs 8, 14 to 17 and 64 to 72)

code to execute a specified functionality for execution on a specified target platform (see abstract; paragraphs 2 and 176; figure 5, step no. 214; figure 6, nos. 252A to G and 260; figure 8, steps no. 310 and 318). It is also disclosed, that the deployed program may be provided in a hardware description language or in the form of text code (see paragraph 120). Furthermore, D1 discloses that individual "nodes" may be configured. Firstly, an "I/O node" is disclosed as having a "default functionality" (or no "defined functionality" at all) until the developer decides to "bind" the node to a particular resource or target platform (see paragraphs 22 and 133) or to indicate the desired operation for that node in the first place (see paragraph 198; see also figure 7, step 302, figure 8, step 316, and figure 16, step 326). It is mentioned that not all nodes may be supported by a chosen target platform (see, e.g., paragraphs 136 and 145). Secondly, the user might be asked to make certain decisions such as to define the "look and feel of the user interface for the graphical program" or "the number or degree of comments" (see paragraph 134).

The decision under appeal

3. The examining division (see decision, page 6, paragraph 1) found claim 1 of then main request to differ from D1 only in that the invention enables users to select implementations according to "certain optimisations", whereas D1 only discloses the selection of a "target platform". This difference was further found not to cause a technical effect or to solve a technical problem (page 6, paragraph 2). Claim 1 of then auxiliary request 1 was found not to comply with Article 123(2) EPC as the term "implementation scope" was not originally disclosed (see the decision, para-

graph bridging pages 7 and 8), and not to be clear due to the phrases "receiving a specification of a designation for an implementation[] of code" and "constructing an intermediate representation, lowering the intermediate representation, and mapping the syntax and semantics of the intermediate representation to the syntax and semantics of the intermediate representation to the syntax and semantics of a target language" (see page 8, items 5.1 and 5.2).

The appellant's position

4. The appellant argued that the invention allowed different parts and/or sections of the given model to be optimized for different needs (see the grounds of appeal, paragraph 2). It also stated that D1 disclosed the generation of a graphical programs from textual input, whereas the invention related to the generation of "code as such" (see page 7, penultimate paragraph, to page 8, paragraph 1; and page 9, third paragraph from the bottom, to page 10, paragraph 1). D1 was thus said to "fail[] to disclose the generation of code that is a programming language or a hardware description language as claimed (see page 9, paragraph below the bullet points), even if the eventual generation of assembly code was conceded to be implicitly disclosed in D1 (see page 10, paragraph 1). Moreover, the appellant took the position that selecting a particular target platform was not comparable to the "designation for performance characteristics or requirement", "for a type of code", or "for a constraint including complying with a coding standard" (page 10, paragraphs 2-3). Accordingly, the appellant concluded that claim 1 of the main request - and, for essentially the same

reasons, that of the auxiliary requests - should be found to be inventive over D1.

5. In its response to the summons and in the oral proceedings, the appellant argued that the invention required the model to be structured in a particular way, in elements, regions and types (see the letter of 15 June 2021, point 21) and the alternative implementations to be structured so that they could be made available via the user interface by means of a "designation". It also made reference to G1/19, saying that the claimed invention is "specifically adapted to objective hardware considerations of an underlying technical system" (point 43 *et seq.*). It further stressed what it considered to be the "gist" of the claimed invention, namely that "various parts of the model can be implemented in a different manner" (see point 48).

Claim construction and original disclosure (Article 123(2) EPC)

6. The appellant indicated (in the annotated version of amended claim 1 of the main request) that the claimed term "implementation scope" was disclosed on page 20, paragraph 2. This arguments apparently rests on the understanding that the "implementation scope" is the "scope the user wants to implement", namely either the element, the element scope or the model scope (see also figure 6, item 608, 610, 614 and 618). The board accepts this argument and considers the term "implementation scope" *per se* not to go beyond the application as originally filed (Article 123(2) EPC).

7. A question arises, however, from the fact that the claims specify an interface which has the user select an element - and which, in response, displays available

implementations for that element - and then an "implementation scope" - and which, in response, displays available implementations for that scope. As for an implementation scope the available implementations may be different than for the element, it would appear that the first displaying step is redundant (see also the summons, point 10).

8. During the oral proceedings, the appellant argued that the second step of displaying designations on the interface was to be understood as an update of the interface, but did not challenge the board's understanding that the display of the interface after the "element" was selected, was redundant in the sense that the eventual code generation was based on the user's selection on the second display of implementation alternatives.

9. The appellant argued, *inter alia*, that D1 did not disclose the generation of textual program code. The board notes, however, that claim 1 of all request specifies the generation of code in a "programming language or a hardware description language". The board also notes that the notion of "programming language" is wide enough to subsume graphical programming languages and does not imply the generation of textual code.

10. Claim 1 of the main request states the interface to "display designations of available implementations" and to receive a "selected of a designation" which would indicate "performance characteristics or requirements", "type of code" or "a constraint" such as a "coding standard". Claim 1 of the auxiliary requests 2-4 specifically requires the implementations to be

designated as "minimiz[ing] power", "area" or "memory usage" or as "maximiz[ing] speed".

10.1 An implementation is then selected based on this indication. As far as the interface is concerned, it is immaterial in what way the available implementations differ and, specifically, whether they differ in the indicated parameters. Moreover, the claims do not specify that the available implementations are analysed in order to determine whether or, if applicable, to what degree they satisfy the indicated requirements. The board takes it that the implementations are labelled, for instance by the provider, as satisfying certain requirements, and that the selection is based on these labels. As far as the interface is concerned, the designations could just as well refer to the language of the comments in the code, the name of the responsible programmer, or merely numbers (see also point 12.2 of the summons).

10.2 As a consequence with regard to inventive step, the board shares the examining division's doubt as regards the technical effect of the claimed invention. In principle, the board considers that the generation of code that is optimal in one of the cited ways (power, area, memory, speed) might be considered a technical effect, although even this statement must be taken with caution due to the inherent vagueness of "optimality" and the difficulty in achieving or guaranteeing it. Irrespective of that, however, any such effect cannot be attributed to an interface that enables to user to select pieces of code according to a characteristics which cannot be relied upon. In this, the board disagrees with the appellant's allegation on page 10,

penultimate paragraph, sentence 1, of its grounds of appeal that the code of the invention is optimized.

Admittance of auxiliary request 2, 3 and 4

11. Auxiliary requests 2, 3 and 4 fall under the provisions of Article 13(2) RPBA2020 according which they shall, in principle, not be taken into account unless there are exceptional circumstances, which have been justified with cogent reasons by the appellant.

12. In view of the discussion the board had - and had to have - with the appellant during oral proceedings with regard to the main request, the board was in a position to give a substantive decision on the auxiliary requests (see below) without compromising procedural economy. Endorsing the reasons of T1294/16 (esp. points 18 and 19) by this board in a different composition, the board thus decides to take into account (or "admit") auxiliary requests 2, 3 and 4 as filed in response to the summons.

Main request

13. When, according to claim 1 of the main request, "the element is selected", an interface displays "available implementations of code comprising a first implementation of code and a second implementation of code". Once the "implementation scope" has been selected, "available implementations" are also displayed. As the implementation scope can be different from the element selected earlier, the available implementations may be different. For instance, where there were two implementations available for the element, only one might be available for the implementation scope. Accepting the appellant's argument that the first

display is redundant for the purpose of the eventual selection of an "implementation of code", it is unclear why that selection should be based, as claimed, on "the first implementation or the second implementation".

13.1 As argued above, the board cannot see that the invention solves a problem of generating optimised code - because neither the generation of optimised code is claimed nor the potential combination of code optimised according to different criteria (see also the summons, point 14) - but considers it to be focused on the interface making certain selections available to the user. This being the focus of the invention, any clarity problem relating to the interface must be taken particularly seriously.

14. The board finds the recited ambiguity to make claim 1 of the main request unclear within the meaning of Article 84 EPC 1973.

First auxiliary request

15. The phrase introduced in claim 1 of the amended first auxiliary request (see point VII above) suffers from a number of clarity issues.

15.1 Firstly, where the rest of claim 1 is formulated, correctly, from the perspective of the computing system performing the individual steps, and mentions the user only indirectly in that the system receives the user's selection, the user is specifically mentioned in the added phrase.

15.2 Secondly, while the rest of claim 1 specifies the user to make selections and the system to generate code, the

added phrase specifies the "user to generate ... code[]".

15.3 Thirdly, and most importantly, the added phrase states the "method [to be] adapted to allow a user" to select "combinations" according to the method "as defined above". What the user is "allowed" to do is, *per se*, a mere result to be achieved, and that it is achieved by (part of) a method "as defined above" is considered to be unclear. More specifically, it is not clear what part of the "above" method steps are meant to be implied by the reference to what is "defined above". The board notes that also the passage in the description which the appellant referred to as original disclosure, namely page 16, second paragraph, also refers only the "the code generation process" which the user might "employ" so as to define "multiple combinations of code". This passage is thus insufficient to overcome the board's clarity concerns.

15.3.1 The appellant argued that the amendment to claim 1 was filed in a reaction to the board's observation, made during the oral proceedings, that claim 1 does not imply that more than one selection is actually made, let alone according to different criteria.

15.3.2 The board notes that this observation was already made in the summons to oral proceedings (see point 14.1). Hence, the appellant has failed to establish exceptional circumstances, let alone given cogent reasons in their support, which could justify that this amendment was filed only in oral proceedings.

15.3.3 Moreover, as explained above, the amendment introduces a number of new problems, at least under Article 84 EPC.

15.4 Accordingly, the board decided not to admit this amendment under Article 13(2) RPBA2020.

Second auxiliary request

16. The second auxiliary request suffers from the same clarity problem that the board established for the main request above. It is therefore found not to comply with Article 84 EPC 1973.

Third and fourth auxiliary requests

17. In the board's judgment, D1 discloses a method of generating code for a model, the model being based on "user's input" and expressed in terms of a graphical or textual model (see paragraph 8 and 64 to 72) and the code being in a programming language or in a hardware description language (see paragraph 120), based on users' selections of implementation alternatives and, optionally, a target platform, which, in the board's judgement, subsumes a choice of target language, (*loc. cit.*, but in particular, figures 7, no. 302, figure 8, no. 316 and figure 16, no. 326). The board considers that graphical programs of the type considered in D1 implicitly has elements ("nodes") and, in fact, types of blocks (e.g. computation blocks, I/O blocks, comparator blocks; see, e.g., figures 10 andd 22A to 22E). D1 discloses that decisions can be made per node (loc. cit.) and may apply to the entire model. It would also appear to be at least obvious that some choice apply to "types of elements": In particular the choice of a "look and feel" (see paragraph 134) should apply alike

to all nodes or elements which relate to the user interface of the generated program.

18. D1 does not disclose the selection of implementation alternatives depending on minimization of power consumption, area or memory use, or on maximization of speed.

18.1 However, as explained above (point 10.1), the invention as claimed does not achieve any such optimization. The claims neither specify how the mentioned optimizations are actually obtained, nor how they might be assessed for consistency and/or compatibility. For the user, the "designations" merely indicate an alleged property of the implementation alternative which they must take for granted and which might be incorrect.

18.2 The board therefore considers that the mentioned optimizations are not actually achieved by the claimed invention and, therefore, cannot be taken into account for assessing inventive step. Effectively, the claimed invention relates to a method of code generation from implementation alternatives, which the user selects via a suitable interface, irrespective of how the implementation alternatives differ from each other. Therefore, the board considers that what the way in which the implementation alternatives are stated, via their "designation", to differ from each other or what they are stated to optimise, cannot contribute to inventive step of the claimed invention.

19. D1 also does not disclose the claimed interface procedure for selecting a part of the model, and amongst the "available implementations" for it.

19.1 The board accepts the appellant's argument that the claim specifies the generation of an interface and, subsequently, its updating, in view of two user's choises made in sequence. As a consequence, the board does not maintain its clarity objection in this regard.

19.2 However, the first display of designations, in response to the selection of an element and before the selection of an (implementation) scope is redundant, because it is modified in an undefined manner by the second display of designations. The board's argument that the first display of designations is redundant was not challenged by the appellant during the oral proceedings (see also point 8 above).

20. The appellant argued during oral proceedings that use of "designations" by the interface as a way of referring to the implementation alternatives required a particular "organisation" of the overall system.

20.1 The board does not agree.

20.2 A system that is to provide implementation alternatives in the first place, which is known from D1 as cited above (point X), there are *a priori* two possibilities: The implementation alternatives can be created on the fly when selected or they can be precomputed. Both alternatives are common place in the art, as they mark the well-known options to trade of computation time against storage requirements. Either way, it is inconceivable that the actual code of the implementation alternatives will be displayed on the screen for the user to choose from. Rather, it is at least obvious that the implementation alternatives are offered for selection "by name" - i.e. by "designation".

20.3 Given the need for a "designation" on the interface, it is evident that the code must be structured accordingly so that, given the designation, the implementation alternative is made available for code generation. How in detail this is organised is not claimed. No inventive structure of the code is, however, required.

20.4 Therefore, the board does not accept that the details of the claimed procedure of displaying information in and having the user select from the claimed interface establishes an inventive step over D1.

20.5 Therefore, claim 1 of the third and the fourth auxiliary requests lacks an inventive step over D1, Article 56 EPC 1973.

Order

For these reasons it is decided that:

The appeal is dismissed.

The Registrar: The Chairman:

A. Voyé Martin Müller

Decision electronically authenticated