UNITED STATES PATENT AND TRADEMARK OFFICE

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

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
14/506,112	10/03/2014	Edward GROSS	6209-40011	7488

25570 7590 10/06/2020
Roberts Calderon Safran & Cole, P.C.
7918 Jones Branch Drive
Suite 500
McLean, VA 22102

EXAMINER	
ST LEGER, GEOFFREY R	

ART UNIT	PAPER NUMBER
2192	

NOTIFICATION DATE	DELIVERY MODE
10/06/2020	ELECTRONIC

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

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

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

docketing@rcsc-ip.com
lgallaugher@rcsc-ip.com
secretaries@rcsc-ip.com

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

Ex parte EDWARD GROSS, DESTRY GUSTIN,
KOMANDUR RAJENDRA KUMAR, PATTABHI RAMA RAO DASARI,
MATTHEW B. TREVATHAN, PRAJAKT DESHPANDE,
RAJ K. KONERU, and SATHYANARAYANA VENNAPUSALA

Appeal 2019-003797
Application 14/506,112
Technology Center 2100

Before JOHN A. JEFFERY, JOHNNY A. KUMAR, and BETH Z. SHAW, *Administrative Patent Judges*.

JEFFERY, *Administrative Patent Judge*.

DECISION ON APPEAL

Under 35 U.S.C. § 134(a), Appellant[1] appeals from the Examiner's decision to reject claims 1–6, 9–15, 17, and 20–25. Claims 7, 8, 16, 18, and 19 were cancelled. We have jurisdiction under 35 U.S.C. § 6(b).

We AFFIRM.

[1] We use the word "Appellant" to refer to "applicant" as defined in 37 C.F.R. § 1.42. Appellant identifies the real party in interest as Kony, Inc. Appeal Br. 2.

STATEMENT OF THE CASE

Appellant's invention is a cross-platform prototyping tool that enables collaboratively designing and developing a single mobile application for multiple platforms with different operating systems. *See generally* Spec. ¶¶ 10–14. Claim 1 is illustrative:

> 1. A method of prototyping an application with a computing device, comprising:
>
> providing a library of pre-defined native design elements which are reproducible in different design layouts on an interface associated with different, predefined platforms; and
>
> providing selecting fields enabling a user to convert, in real time without recompiling any design code, a first design layout comprising one or more native design elements which fits within an interface associated with a first platform of the different, predefined platforms to a second design layout comprising the one or more native design elements which fits within an interface associated with a second platform of the different, predefined platforms, the second platform having a different operating system than an operating system of the first platform, and which is in proportion to the first design layout by using the pre-defined native design elements for the second platform in the library,
>
> wherein the converting comprises automatically reformatting the one or more design elements to fit within a virtual interface associated with the second platform without the need to reformat or rewrite any code.

THE REJECTIONS

The Examiner rejected claims 1–6, 12, 13, 21, and 25 under 35 U.S.C. § 103 as unpatentable over Mukkamala (US 2011/0154287 Al; published

June 23, 2011) and Seven (US 2013/0205277 Al; published Aug. 8, 2013). Ans. 3–10.[2]

The Examiner rejected claims 9–11, 14, 15, 17, 20, and 24 under 35 U.S.C. § 103 as unpatentable over Mukkamala, Seven, and Hsu (US 8,938,679 Bl; issued Jan. 20, 2015). Ans. 10–22.

The Examiner rejected claim 22 under 35 U.S.C. § 103 as unpatentable over Mukkamala, Seven, Lappas (US 8,046,694 Bl; issued Oct. 25, 2011), and Johnson (US 2015/0346982 Al; published Dec. 3, 2015). Ans. 22–24.

The Examiner rejected claim 23 under 35 U.S.C. § 103 as unpatentable over Mukkamala, Seven, Hsu, Lappas, and Johnson. Ans. 25–26.

THE OBVIOUSNESS REJECTION OVER MUKKAMALA AND SEVEN

The Examiner finds that Mukkamala prototypes applications by enabling a user to convert a design layout that fits within an interface associated with a first platform to a design layout that fits within an interface associated with second platform having a different operating system, where the conversion automatically reformats the layout's design elements to fit within the second platform's virtual interface without needing to reformat or rewrite any code. Ans. 3–5, 27–30, 34–35, 37. Although the Examiner acknowledges that Mukkamala does not enable this conversion in real time without recompiling design code and provide selecting fields in connection

[2] Throughout this opinion, we refer to (1) Appeal Brief filed October 22, 2018 ("Appeal Br."); (2) the Examiner's Answer mailed February 11, 2019 ("Ans."); and (3) the Reply Brief filed April 11, 2019 ("Reply Br.").

with this conversion, the Examiner cites Seven for teaching this feature in concluding that the claim would have been obvious. Ans. 5–6, 30–34, 36, 41.

Appellant argues that the cited prior art fails to disclose (1) the recited library of pre-defined native design elements that are reproducible in different platform interface layouts, and (2) converting a first platform's layout to a layout in a second platform with a different operating system without needing to recompile any design code as claimed. Appeal Br. 4–10; Reply Br. 2–12. Appellant adds that Mukkamala does not automatically reformat the design elements to fit in the second platform's interface without needing to reformat or rewrite any code as claimed, but rather requires the developer to do so. Appeal Br. 12–13; Reply Br. 6–8. Appellant argues other recited limitations summarized below.

ISSUES

Under § 103, has the Examiner erred by finding that Mukkamala and Seven collectively would have taught or suggested:

(1)(a) providing a library of pre-defined native design elements that are reproducible in different platform interface layouts; and (b) providing selecting fields enabling a user to convert a first design layout that fits within an interface associated with a first platform to a second design layout that fits within an interface associated with second platform having a different operating system, where the conversion automatically reformats the layout's design elements to fit within the second platform's virtual interface without needing to reformat or rewrite any code as recited in claim 1?

(2) the limitations recited in claim 25?

ANALYSIS

Claims 1–6, 12, 13, and 21

On this record, we see no error in the Examiner's reliance on Mukkamala and Seven for collectively teaching or suggesting the elements recited in claim 1. First, we see no error in the Examiner's finding that Mukkamala at least suggests providing a library, namely a screen design palette, of pre-defined native design elements, namely controls, that are reproducible in different platforms' interface layouts consistent with the Examiner's mapping. *See* Ans. 3.

Mukkamala's Figure 3 shows the process for designing and generating a mobile device application. As shown in that figure, not only can a mobile application screen be selected from a palette of pre-defined screens in step 306, but a screen design palette can also be used to add controls to the screen in step 312. *See* Mukkamala ¶¶ 44, 47. As the Examiner explains, these control-based elements are pre-defined because they already exist in the palette for transfer to the application screens. Ans. 27. So even assuming, without deciding, that native design elements must be stored in the recited library as Appellant contends (Appeal Br. 6; Reply Br. 10), Mukkamala at least suggests as much given the palette of such elements from which the designer can select in designing an application. *See* Mukkamala ¶¶ 44, 47.

Nor are we persuaded of error in the Examiner's reliance on Mukkamala for at least suggesting enabling a user to convert a first design layout that fits within a first platform's interface to a second design layout that fits within a second platform's interface associated with a different operating system. Ans. 3–5, 27–30, 34–35, 37. As shown in the exemplary platform-specific views of user interface controls in Mukkamala's Figure 6, the four radio buttons are shown as a vertical list in the BLACKBERRY device 610, but are depicted as a horizontal array of contiguous rectangular boxes on the IPHONE device 612 as the Examiner indicates. *See* Ans. 28, 37.

This conversion of the BLACKBERRY layout with its respective native design elements, namely the vertically-spaced radio button controls, to the IPHONE layout with its respective native design elements, namely the horizontal array of contiguous rectangular boxes, in Mukkamala's Figure 6 is made possible by the developer merely selecting a different device or platform. *See* Mukkamala ¶ 68. Notably, this selection enables the developer to change the view dynamically without having to first execute the application on a simulator. *See id.*

Although Mukkamala does not state explicitly that this "dynamic" change is made automatically, Mukkamala at least suggests as much, or that such an automatic change would have been at least an obvious variation, particularly given the functionality of the application designer associated with the developer's selection. *See* Mukkamala ¶ 68. That this selection enables the developer to change the view dynamically *without having to first execute the application on a simulator* only underscores Mukkamala's suggestion that at least some of the functionality associated with this change is automatic or at least an obvious variation. *See id.*

We reach this conclusion despite the developer's involvement in this dynamic view change, for the recited term "automatically reformatting" does not preclude at least some manual steps, so long as other steps are automatic. *See CollegeNet, Inc. v. ApplyYourself, Inc.*, 418 F.3d 1225, 1235 (Fed. Cir.

2005). Moreover, merely replacing manual activity with automatic means to accomplish the same result is an obvious improvement. *See In re Venner*, 262 F.2d 91, 95 (CCPA 1958). Nor has Appellant shown that performing the recited reformatting functions automatically in lieu of at least some manual interaction would have been uniquely challenging or otherwise beyond the level of ordinarily skilled artisans. *See Leapfrog Enters., Inc. v. Fisher-Price, Inc.*, 485 F.3d 1157, 1161–62 (Fed. Cir. 2007) ("Applying modern electronics to older mechanical devices has been commonplace in recent years.").

Appellant's contention, then, that the developer's dynamically changing the view in Mukkamala's paragraph 68 is ostensibly a manual—not automatic—operation that requires the developer to reformat or rewrite code (Appeal Br. 10–11; Reply Br. 6–8, 13–14) is not only speculative and unsubstantiated on this record, the recited automatic reformatting does not preclude at least some manual steps as noted above. Moreover, given Mukkamala's code listing in paragraph 68 that renders the radio buttons in the BLACKBERRY device in Figure 6, ordinarily skilled artisans would understand that there would also be existing code that renders the commensurate functionality associated with horizontal array of contiguous rectangular boxes for the IPHONE device in that figure. By selecting either of these platforms, ordinarily skilled artisans would understand from Mukkamala's layout conversion functionality that the code rendering the respective control-based design elements would not be reformatted or rewritten, but merely *selected* consistent with the developer's selection. *See* Mukkamala ¶ 68; Fig. 6. *Accord* Ans. 5, 35 (noting that because the developer can dynamically change (i.e., automatically reformat) the view of the user interface control by selecting a different device or platform, reformatting or rewriting code is unnecessary).

To be sure, Mukkamala does not state explicitly that this layout conversion occurs in real time without recompiling any design code or that selecting fields are provided to enable this conversion. Nevertheless, we see no error in the Examiner's reliance on Seven for teaching these features in concluding that the claim would have been obvious. *See* Ans. 5–6, 30–34, 36, 41.

First, Appellant does not persuasively rebut the Examiner's mapping the functionality associated with Seven's device selection control 562c in Figures 5D and 5E to the recited selecting fields, particularly given the associated dialog box presented to the user and its associated fields upon selecting this control. *See* Ans. 5, 36, 38–39 (noting that selections made via this selection-control-based dialog box would be represented by a field).

Nor does Appellant persuasively rebut the Examiner's reliance on the functionality associated with Seven's smart phone and personal digital assistant (PDA) simulations in Figures 5D and 5E for at least suggesting enabling a user to convert layouts for different platforms in real time without recompiling any design code as claimed. *See* Ans. 5–6, 30–34, 36, 41. We reach this conclusion noting, as does the Examiner, that Seven's conversion functionality is based on rendering *non-compiled* presentation files that, according to paragraph 34, can be in (1) mark-up languages such as HyperText Markup Language (HTML); (2) *interpreted* languages such as JavaScript; and (3) style sheet languages such as Cascading Style Sheets (CSS). That these presentation files are *interpreted* as emphasized above only bolsters the Examiner's finding that Seven's presentation files are interpreted—not compiled—for rendering within a selected device simulation. *See* Ans. 34.

To be sure, Seven's conversion occurs without *compiling* any design code as the Examiner indicates, despite the claim reciting that conversion occurs without *recompiling*. *See* Ans. 5 (noting that Seven allows converting a first arrangement of controls 558 in smart phone display area 555 to a different arrangement without *compiling* associated presentation files). Nevertheless, if conversion occurs without compiling, it also occurs without recompiling. *Accord* Ans. 41. That is, recompiling requires compiling code in the first instance so that it can be compiled again or *re*compiled. *Accord* Reply Br. 11 (noting this point). But code that is not compiled is, by its very nature, not *recompiled* because it was not compiled in the first place. Therefore, because Seven's layout conversion occurs without compiling, it also occurs without recompiling. Appellant's arguments to the contrary (Appeal Br. 6–7, 9, 11; Reply Br. 11) are unavailing and not commensurate with the scope of the claim. Moreover, because Seven interprets code, such as JavaScript, to effect this conversion as noted in paragraph 34, Seven at least suggests a real-time conversion. *Accord* Ans. 5–6, 34, 41 (noting that ordinarily skilled artisans would understand that an interpreter's operations differ from those of a compiler).

Therefore, we are not persuaded that the Examiner erred in rejecting claim 1, and claims 2–6, 12, 13, and 21 not argued separately with particularity.

Claim 25

We also sustain the Examiner's rejection of claim 25 reciting, in pertinent part, a processor executes program instructions to allow the designer to revise or modify the design layout based on the comments without the need to recode any design layout by dropping and dragging the native design elements from the library to the design layout.

As a preliminary matter, we note a key inconsistency in the claim. Claim 25 depends from method claim 21 that depends from method claim 1. Claim 25, however, further limits "[t]he computer system of claim 21" despite claim 21 being a method claim—a different category of statutory subject matter. *See* 35 U.S.C. § 101. Nor is there antecedent basis for "*the* program instructions," "*the* designer," or "*the* comments" in claim 25 or in claims 1 and 21 from which claim 25 depends. Although claim 25 was not rejected under § 112 as indefinite despite these defects, we nonetheless leave this issue to the Examiner to consider should prosecution continue after this decision.

Turning to the rejection, the Examiner finds that Mukkamala's paragraphs 46, 47, and 68 teach a processor executing program instructions without the need to recode any of the design layout by dropping and dragging native design elements from the library to the design layout. Ans. 9–10.

Appellant does not persuasively rebut these findings, let alone the Examiner's construing the phrase "to allow the designer to make revisions or modifications to the design layout based on the comments" as merely an intended use of the recited processor execution and, therefore, does not further limit that execution. Ans. 10. Rather, Appellant merely cites the

Specification's paragraphs 58 and 59, and alleges that the "highly convenient" arrangement for communicating between a client and designer without needing to recode any of the design layout in claim 25 is lacking from the cited prior art. Appeal Br. 16.

This contention is unpersuasive, for not only does Appellant fail to address—let alone persuasively rebut—Mukkamala's particular passages cited by the Examiner for teaching this feature, Appellant does not explain why the Examiner's findings are erroneous apart from mere conclusory statements. *Accord* Ans. 48 (noting this point).

Although Appellants argue for the first time on pages 21 and 22 of the Reply Brief regarding *Hsu*'s alleged shortcomings in connection with the rejection of claim 25, such arguments were not raised in the Appeal Brief and are, therefore, waived as untimely. *See* 37 C.F.R. § 41.41(b)(2) (2018). Nor has good cause been shown to raise these new arguments in the first instance in the Reply Brief.

Nevertheless, even if these belated arguments regarding Hsu's alleged shortcomings in connection with the rejection of claim 25 were timely presented—which they were not—Appellant's arguments are not germane to the Examiner's rejection that does not rely on Hsu, but rather only Mukkamala and Seven. *See* Ans. 3, 9–10. Nor do these untimely arguments squarely address—let alone persuasively rebut—the Examiner's construing the phrase "to allow the designer to make revisions or modifications to the design layout based on the comments" as merely an intended use of the recited processor execution as noted above. *See* Ans. 10.

Therefore, we are not persuaded that the Examiner erred in rejecting claim 25.

THE REJECTION OVER MUKKAMALA, SEVEN, AND HSU

Claims 14, 15, and 17

We also sustain the Examiner's rejection of independent claim 14 reciting, in pertinent part, (1) providing selecting fields enabling a user to switch between different platforms having different operating systems from one another to visualize the design layout with native design elements in the different platforms in real time without recompiling any design code; (2) transferring the design layout with the native design elements to the second computing device to render the native design elements natively as a functional prototype application in its interface; (3)(a) tagging comments to native design elements, and (b) collaboratively sharing the tagged comments associated with the native design elements to different computing devices including a native platform; (4) transferring the tagged comments in real time between the native platform and second computing device using a server to tag the tagged comments to the native design elements on the native platform on the second computing device, where changes can be made to the functional prototype application on the second computing device based on the tagged comments without the need to recode the native design elements.

Regarding elements (1) and (2) above, we see no error in the Examiner's reliance on Mukkamala and Seven for collectively teaching those features for the reasons previously discussed. *See* Ans. 13–15.

Nor do we see error in the Examiner's reliance on Hsu for at least suggesting (1) tagging comments to native design elements; (2) collaboratively sharing the tagged comments associated with the native design elements to different computing devices including a native platform;

and (3) transferring the tagged comments in real time between the native platform and second computing device using a server to tag the tagged comments to the native design elements on the second computing device's native platform. Ans. 15–17, 42–48.

As explained in the Abstract, Hsu's system enables collaboration on specifying an interactive graphical design including, among other things, a note interface that displays a note field for accepting a user's text string. Hsu's system also includes a discussion interface that not only displays the user's text string as a note, but also accepts comments from a second user regarding the note. Hsu Abstract.

As shown in the design environment 200 in Hsu's Figure 2a, a user can add not only add widgets, such as buttons, text fields, display windows, etc., to design 201 by dragging and dropping widgets into the design interface 202, but also link widgets to a text-based note 208. Hsu col. 3, l. 58 – col. 5, l. 10. Moreover, additional notes can be added via field 209. Hsu col. 5, ll. 11–16.

As shown in Hsu's Figure 2b, the design is rendered in design player 210 that includes (1) link 216 between note 213 and text field 205; and (2) discussion interface 212 that not only allows viewers of the design to add notes to discuss the design, but also includes live chat interface 214 that allows multiple viewers to discuss the design. Hsu col. 6, ll. 61–65; col. 9, ll. 39–67.

This note-linking and discussion functionality at least suggests (1) tagging comments to native design elements, namely widgets, and (2) collaboratively sharing the tagged comments to different computing devices, namely those used by the viewers of the design, for discussion.

Hsu's functionality also at least suggests transferring the tagged comments between a native platform and a second computing device using a server to tag the tagged comments to the native design elements on the second computing device's native platform. That viewers can access the design via user interface system 405 on their respective work stations or computing devices as the Examiner indicates (Ans. 43–44) at least suggests that tagged comments are transferred between these devices' platforms, and that a server, such as processing system 401, is used to achieve this end. *See* Ans. 43–44 (citing Hsu col. 6, ll. 61–65; col. 7, ll. 7–13, 32–35; col. 13, ll. 42–43, 53–58; Fig. 4).

Notably, notes can appear *in real time* in Hsu's design environment and player to allow viewers and users to modify and comment on the design *simultaneously* as one stakeholder operates the design tool, while the other views the design in the player. Hsu col. 11, ll. 7–13. Moreover, stakeholders can receive *instant messages* regarding changes *as they occur*. Hsu col. 12, ll. 32–34.

This real-time functionality at least suggests transferring tagged comments in real time between devices' platforms for simultaneous viewing and collaboration as changes occur. Appellant's contention that Hsu's emailed comments are not in real time (Appeal Br. 15) is unavailing, for this argument is not germane to Hsu's *real-time* comment functionality noted above. Moreover, Hsu's functionality also enables changing the design on the second computing device based on the tagged comments by, among other things, adding widgets by dragging and dropping them into the design interface as noted in column 3, lines 58 to 67—changes that do not require recoding these native design elements as the Examiner indicates. *See* Ans. 16, 44–45.

To be sure, Hsu does not state explicitly that the computing devices' platforms have different operating systems as claimed. But the Examiner's rejection is not based solely on Hsu, but rather the collective teachings of Mukkamala, Seven, and Hsu. That is, the Examiner relies on Mukkamala and Seven for teaching the first three clauses of claim 14, including (1) providing selecting fields enabling a user to switch between different predefined platforms with different operating systems in real time without recompiling any design code, and (2) transferring the design layout to the second computing device to render the native design elements as claimed. *See* Ans. 13–15. The Examiner cites Hsu, however, merely to show that the recited comment tagging, sharing, and transfer limitations in claim 14 are at least suggested by Hsu in concluding that the claim would have been obvious over the cited references' collective teachings. *See id.* Therefore, Appellants' arguments regarding Hsu's individual shortcomings regarding transferring the design layout between the platforms' different operating systems (Appeal Br. 11–15; Reply Br. 17–19) do not show nonobviousness where, as here, the rejection is based on the cited references' collective teachings. *See In re Merck & Co.,* 800 F.2d 1091, 1097 (Fed. Cir. 1986).

Therefore, we are not persuaded that the Examiner erred in rejecting claim 14, and claims 15 and 17 not argued separately with particularity.

Claim 20

We also sustain the Examiner's rejection of independent claim 20. Ans. 18–21, 49–53. First, despite Appellant's arguments to the contrary

(Appeal Br. 17–20), we are unpersuaded of error in the Examiner's reliance on the collective teachings of Mukkamala, Seven, and Hsu for at least suggesting the disputed limitations of claim 20, including those limitations commensurate with those recited in independent claims 1 and 14, for the reasons discussed previously.[3] Moreover, Appellant's contention that Hsu fails to teach or suggest "transferring the tagged comments in real-time between the native platform and the second computing device using the server to tag the tech comments to the native design elements" (Appeal Br. 18) is unavailing, for claim 20 recites no such limitation. *Accord* Ans. 50 (noting this discrepancy). Nevertheless, we are unpersuaded in the Examiner's reliance on Hsu for teaching the recited comment-based limitations, particularly when considered in light of Mukkamala and Seven for the reasons previously discussed.

We are also unpersuaded of error in the Examiner's reliance on Hsu for at least suggesting that *when* a third party user selects comments made by the user regarding the specific native design elements, after the comments are shared, the third party user will automatically be brought to the specific native design elements which the comments pertain to. Ans. 21, 52 (citing Hsu col. 9, ll. 39–41, 54–65; Fig. 2b).

Our emphasis on the term "when" underscores that the limitation specifies a condition under which the recited automatic functionality occurs, namely *when* a third party user selects the comments. Although conditional

[3] We note in passing that no antecedent basis exists for "the tagged comments" in claim 20. We nevertheless leave the question of whether this defect renders the claim indefinite to the Examiner to consider should prosecution continue after this decision.

limitations need not be satisfied to meet method claims, that is not the case here, where the claim recites an apparatus whose structure performs a function that occurs only if a condition is satisfied. *See Ex parte Schulhauser*, No. 2013-007847, 2016 WL 6277792 (PTAB Apr. 28, 2016) (precedential); *see also* MANUAL OF PATENT EXAMINING PROCEDURE (MPEP) § 2111.04(II) (9th ed. Rev. 10.2019, June 2020) (citing *Schulhauser*).[4] That is, for apparatus claims, the claim still requires structure for performing the function should the condition occur.

Nevertheless, Hsu at least suggests the recited functionality that automatically brings the user to the specific native design elements to which the comments pertain (1) when the user selects the comments, and (2) after the comments are shared. As Hsu explains, a visual indicator, such as a highlighted outline around a note, indicates that the widget to which the note applies is not displayed in the rendered design. Hsu col. 9, ll. 60–63. But in these instances, a hyperlink can be automatically added to the note that *links to a rendering in which the widget appears*. Hsu col. 9, ll. 63–65.

The import of this discussion is that when the user selects hyperlinks in the shared notes or comments, the user will be automatically be brought to the specific native design elements or widgets to which the comments pertain. Although some manual operation is required to effect this functionality, namely selecting the hyperlink, this hyperlinking functionality nevertheless automatically brings the user to the associated widget or native design element by merely clicking the hyperlink.

[4] Although the limitations at issue in *Schulhauser* were rendered conditional by reciting the term “if,” we discern no meaningful distinction between the recitation of “if” and “when” in this context.

Furthermore, bringing the user to the specific native design elements *automatically* as claimed does not preclude at least some manual steps, so long as other steps are automatic. *See CollegeNet*, 418 F.3d at 1235. Moreover, merely replacing manual activity with automatic means to accomplish the same result is an obvious improvement. *See Venner*, 262 F.2d at 95. Nor has Appellant shown that performing the recited functionality automatically in lieu of at least some manual interaction would have been uniquely challenging or otherwise beyond the level of ordinarily skilled artisans. *See Leapfrog*, 485 F.3d at 1161–62 ("Applying modern electronics to older mechanical devices has been commonplace in recent years.").

Appellant's contention that Hsu does not teach or suggest the recited automatic functionality because additional action is required by the third party user (Reply Br. 26–27) is unavailing and not commensurate with the scope of the claim, which does not preclude at least some manual activity. On this record, Hsu at least suggests the recited automatic functionality.

Therefore, we are not persuaded that the Examiner erred in rejecting claim 20.

Claims 9–11 and 24

We also sustain the Examiner's rejection of claim 24 reciting that the tagged comments are made by a first user, and when a designer or second user selects the tagged comments made by the first user regarding the particular design element after the comments are transferred, the designer or second user will automatically be brought to the particular design element to which the tagged comments pertain. *See* Ans. 22, 52–54. For the reasons indicated previously with respect to claim 20 that recites commensurate limitations, and despite Appellant's arguments to the contrary (Appeal Br. 21), we are unpersuaded of error in the Examiner's rejection of claim 24, and claims 9–11 not argued separately with particularity.

THE REJECTION OVER MUKKAMALA, SEVEN, LAPPAS, AND JOHNSON

We also sustain the Examiner's rejection of claim 22 over Mukkamala, Seven, Lappas, and Johnson. Ans. 22–24, 54. Claim 22 recites that the selecting fields include (1) a first scrollable selecting field to select among different operating systems, and (2) a second scrollable selecting field to select among different device types with different resolutions.

We find unavailing Appellant's contention that the scrollable fields 305a and 305b in Figure 3 of the present application permit a finer degree of control that is possible with Seven's device selection control 562c, and that neither Lappas nor Johnson cure Mukkamala's and Seven's previously-noted deficiencies. *See* Appeal Br. 22. Notably, Appellant's contentions in the Appeal Brief do not squarely address—let alone persuasively rebut—the Examiner's reliance on (1) Lappas's "OS" field 415 in Figure 5, and (2) Johnson's alternate device selection drop-down list in paragraph 67 and Figure 12 for teaching scrollable fields for selecting among different operating systems and device types, respectively, and that providing such fields in the Mukkamala/Seven system would have been at least an obvious variation. *See* Ans. 23–24.

Although Appellant contends for the first time on pages 27 and 28 of the Reply Brief that using drop-down menus to select operating systems or target platforms in Lappas and Johnson, respectively, does not teach or suggest the recited selecting fields' role when considered in conjunction with the elements recited in claims 1 and 14, this argument was not raised in the Appeal Brief and is, therefore, waived as untimely. *See* 37 C.F.R. § 41.41(b)(2). Nor has good cause been shown to raise this new argument in the first instance in the Reply Brief.

Nevertheless, even if this belated argument regarding Lappas's and Johnson's alleged shortcomings in connection with the rejection of claim 22 was timely presented—which it was not—providing dedicated scrollable selecting fields to select among different operating systems and device types, respectively, as the Examiner proposes uses prior art elements predictably according to their established functions—an obvious improvement. *See KSR Int'l Co. v. Teleflex, Inc.*, 550 U.S. 398, 417 (2007). To the extent that Appellant contends that Lappas's and Johnson's drop-down selection lists are somehow not scrollable selecting fields (*see* Appeal Br. 22; Reply Br. 27–28), such arguments are unavailing and not commensurate with the scope of the claim.

Therefore, we are not persuaded that the Examiner erred in rejecting claim 22.

THE REJECTION OVER MUKKAMALA, SEVEN, HSU, LAPPAS, AND JOHNSON

We also sustain the Examiner's obviousness rejection of claim 23 over Mukkamala, Seven, Hsu, Lappas, and Johnson. Ans. 25–26. Because this rejection is not argued separately with particularity, we are not persuaded of error in this rejection for the reasons previously discussed.

CONCLUSION

In summary:

Claims Rejected	35 U.S.C. §	Reference(s)/ Basis	Affirmed	Reversed
1–6, 12, 13, 21, 25	103	Mukkamala, Seven	1–6, 12, 13, 21, 25	
9–11, 14, 15, 17, 20, 24	103	Mukkamala, Seven, Hsu	9–11, 14, 15, 17, 20, 24	
22	103	Mukkamala, Seven, Lappas, Johnson	22	
23	103	Mukkamala, Seven, Hsu Lappas, Johnson	23	
Overall Outcome			1–6, 9–15, 17, 20–25	

TIME PERIOD FOR RESPONSE

No time period for taking any subsequent action in connection with this appeal may be extended under 37 C.F.R. § 1.136(a)(1).

AFFIRMED