DisputeSoft
Software Dispute Experts

Patent Litigation: An Introduction to Patent Claims, "Limitations," Infringement, and Invalidity -- Part Five

Andrew Schulman

Senior Software Litigation Consultant, DisputeSoft

SoftwareLitigationConsulting.com

In this six-part series, we've discussed how the actionable part of a patent is its "claims" (Part 1); how claims are made of "limitations" which are generally those elements of an invention necessary to distinguish it from earlier technology ("prior art") (Part 2, which also discussed "claim construction," *i.e.*, interpreting the meaning of terminology appearing in patent claims); how to use patent claim limitations while investigating patent infringement (Part 3); and how to structure comparisons of single claim limitations, including breaking down the limitation into subparts and utilizing the function/way/result test for equivalence (Part 4).

In this Part 5, we'll look at:

- The concept of patent invalidity: while patents granted by the government (in the U.S., the PTO) are presumed valid, and while this presumption of validity forms a large part of the value of a patent, nonetheless this presumption can be rebutted (with a sufficiently high level of proof), and claims of the patent found to be invalid;
- Patent claims can be invalidated by showing: anticipation (lack of novelty), "obviousness" (a term of art we'll discuss in detail), lack of "enablement" (failure to sufficiently disclose how to make and use a claimed invention), and the on-sale and public-use bars, among others;
- Invalidating a patent claim by showing anticipation (lack of novelty) depends on the "prior art";

- Investigating prior art for a software patent claim: what counts as "prior art," and how a patent claim is compared with prior art, to determine if the claim is invalid for lack of novelty;
- Checking prior art already found by the PTO (while it was examining a patent application), as reflected in the patent's "file wrapper";
- Performing a limitation-by-limitation comparison of a specific prior-art reference with a patent claim (here, claim 1 of the "facade-server" software patent that we've been using as a running example);
- Using obviousness as another way to invalidate a patent: looking for the ordinary non-inventive skilled practitioner's "motivation to combine" multiple prior-art references;
- How a patent owner can respond to assertions that its patent claim is invalid for anticipation or obviousness;
- Some closing thoughts on patents as "shaky" property, due to the almost ever-present risk of patent invalidation.

Patent invalidity: pulling the rug out from under P's property

In Part 4, we looked at an example of locating and analyzing infringement of a software patent claim from patent 7,472,398 relating to a so-called "facade-server" (which the patent owner defined as a web server without a network connection), so that a web browser on the same computer can be used as a display engine or GUI front-end for local applications.

The discussion included problems the plaintiff (P) might encounter during its infringement analysis; such problems would be grist for the defendant (D)'s *non*-infringement analysis. P may have a good sense of how D is going to respond when it's sued for infringement, based on problems P itself had in fitting D's product to P's claim.

In addition to asserting its own non-infringement, D will usually also argue that P's patent is not valid in the first place. In other words, not only "I don't infringe," but also "and even if I did, it doesn't matter, because your patent is no good." An analogy is "P says I trespassed on his land, but I didn't step on that land, and it doesn't even matter if I did, because P doesn't own that land,

and/or possibly no one owns that land." D tries to both (a) deflect P's punches, and (b) pull the rug right out from under P. This is a serious risk to P because if the patent is invalidated in the case P v. D1, it likely would be invalid for purposes of any future P v. D2, P v. D3, etc. P effectively has to win on invalidity each time. (To the astute reader saying, "wait a minute, it's patent claims that are invalidated, not necessarily the entire patent," hold that thought, we'll get to it).

That a patent, granted by a government agency (in the U.S., the PTO) after a lengthy examination, might nonetheless be invalid, may sound surprising. Patents are a form of property (see Part 1). While government of course takes away property in situations such as criminal forfeiture, that typically involves a forced *transfer* of a given piece of property. That a patent itself may be invalidated, *i.e.*, wiped out, seems different, even though the patent only came about because of a government grant, and even though in some cases the PTO is "merely" (though no small thing to the patent owner) reconsidering its own earlier decision (see the *Oil States* case, discussed below in the section on "Shaky property").

We've noted earlier in this series that patents are presumed valid, and that indeed this presumption of validity forms a large part of the value of a patent (as distinct from the value of the underlying invention). The presumption of validity is what allows a patent owner to sue for infringement, at least when coupled with a plausible basis (see Part 6) for asserting that the defendant infringes the patent, and allows P to do so without having to re-prove up-front that its claimed invention meets the requirements for patentability (novelty, non-obviousness, sufficient disclosure, and so on). The patent grant itself establishes those things.

Or, rather, establishes a *presumption* of those things. This presumption can be rebutted. However, for D to rebut the presumption of validity during litigation and show that P's government-granted patent claim is invalid, D must do so using "clear and convincing evidence." This is a higher standard than the "preponderance of the evidence" (*i.e.*, a bit more than 50%) standard under which P must show D's infringement, though it is lower than the "beyond a reasonable doubt" standard in criminal law.

In other words, D has an uphill battle in trying to show that P's patent is invalid, and a heavier burden than P carries in showing that D infringes. At the same time, about 40% of patents whose validity is challenged during infringement litigation in federal court do end up being invalidated

in whole or in part (see Professor Thomas Cotter's excellent recent book, *Patent Wars: How Patents Impact Our Daily Lives*).

That "in part" is important: while I've been referring to "invalid patents," in fact invalidity, like infringement, must be generally analyzed on a per-claim basis; some patent claims could be invalid without rendering all claims invalid. In fact, a dependent claim may still be valid, even if its underlying independent claim is invalid (see box below).

How dependent claims intersect with infringement and invalidity

Most patents contain a range of broader and narrower independent and dependent claims. Perhaps surprisingly, a dependent claim may still be valid, even if its underlying independent claim is invalid. This is in contrast to infringement, where (as one might expect) a dependent claim can only be infringed if the underlying independent claim is also infringed.

While this difference -- dependent can be valid even if independent is invalid, while dependent cannot be infringed unless independent is also infringed -- sounds strange at first, it does make sense:

For example, if dependent claim 2 has an additional limitation C, not found in independent claim 1 containing A+B, infringement requires doing all three A+B+C. However, the combination A+B+C in claim 2 could still be novel (not anticipated in the prior art), even if A+B in claim 1 wasn't novel. For this reason, P may want to try to prove infringement of narrower/safer claim 2 as well as of broader/riskier claim 1. Having a range of claims, from broader/riskier to narrower/safer, gives the patent owner options in how to steer between the dangers of its own invalidity on the one hand, and D's non-infringement on the other hand.

Patent claims can be invalidated in a number of ways: by showing anticipation (lack of novelty); "obviousness" (a term of art we'll discuss in detail towards the end of this Part 5); lack of "enablement" (failure to sufficiently disclose how to make and use a claimed invention); P's violation of the so-called on-sale or public-use bars (basically, P waited too long to apply for its patent, after already selling or publicly using the claimed invention; see Helsinn v. Teva); for ineligible "subject matter" (see below briefly on *Alice*); and for other reasons.

Further, an entire patent might also be rendered "unenforceable" if it turns out that the patent owner didn't disclose to the PTO, during the examination of its patent application, known material information (such as relevant prior art P knew of). Unenforceability is distinct from

invalidity: the patent is still valid, but P can't do much with it unless the underlying problem is cured. P has no obligation to proactively search for prior art, but if P did know of it (see "The ostrich position" below), and it turns out to matter to the validity determination, this can constitute "inequitable conduct" before the PTO.

The ostrich position

A patent owner carries some risk if its engineers are familiar with relevant prior art that isn't disclosed to the PTO. In part for this reason (also to avoid what is called "willful" infringement), some companies actively *discourage* their engineers from researching patent disclosures; their knowledge of a patent's existence would be imputed to the entire organization. Of course, this ostrich position -- incentivized by the otherwise-reasonable inequitable-conduct rule -- directly contradicts two key purposes of patents, to disclose inventive technical knowledge to relevant technical communities, and to discourage "not invented here" ignorance of the prior art.

Investigating prior art for a software patent claim

We now turn to the task of locating and analyzing prior art that might help D invalidate P's patent claim, by showing its lack of novelty (anticipation). A patent can of course only be valid if the underlying invention was new, *i.e.*, not anticipated in the prior art.

By "prior art," we here mean evidence that would help D show that P's invention is not, in fact, an invention, but instead something that was already known, or was an "obvious" extension, at the time P applied for its patent.

The evidence might be a slam-dunk, showing earlier disclosure of each and every limitation of the patent claim, or it might only disclose some limitations, and -- in order to invalidate the patent claim -- need to be combined with other evidence that supplies the missing limitations. The combination of multiple prior-art references requires an analysis of "obviousness," which we'll discuss after first working through a single-reference example, as part of anticipation. Discussing anticipation before obviousness matches real-world patent practice, because obviousness is D's fallback second choice, in somewhat the same way that (as discussed in Part 4) equivalence is P's fallback second choice if it has trouble showing literal infringement.

Along the way, we'll also see opportunities for P to try to show validity, by distinguishing its patent claim from the prior-art references found by D. P has no *up-front* obligation in litigation to re-prove the validity of its already-granted patent. *That's precisely what the patent grant is for*.

P's obligation, with regard to the validity of the patent it asserts against D, is to rebut any specific invalidity arguments that D has made. Similarly, D has no proactive obligation to prove its non-infringement in some global sense; it need only respond to P's specific infringement contentions.

Alice and the validity of software patents

Since we're talking about invalidating a software patent claim, some readers may be thinking, "Wait a minute, didn't that *Alice* case invalidate *all* software patents?" The US Supreme Court's 2013 decision in *Alice v. CLS Bank* had to do with challenges to patent validity under 35 USC 101 on "subject matter" (generally speaking, is software the sort of thing that could be patented?), whereas here we're focusing on 35 USC 102 on novelty and anticipation (and at the end of this article, 35 USC 103 on obviousness), but certainly, *Alice* has had a large impact on the validity of software patents. In the roughly two-year period following *Alice*, of about 15,000 claims that were challenged on §101 grounds, about two-thirds of these claims (10,000) were found invalid. But this also means that one-third survived serious challenge, and the "death of software patents" has been (as Mark Twain said of his own death) greatly exaggerated. There are sufficient software patent §101 post-*Alice* eligibility findings that guidelines can be derived by carefully comparing these findings (search, *e.g.,* for Enfish, DDR, Finjan, MCRO) to the admittedly larger number of invalidations; see the PTO's 2019 guidelines. In a later article, I will discuss §101 obstacles to software-patent validity, but for now, let's return to trying to invalidate a software patent claim using §102 anticipation.

What counts as prior art

The reader probably has a general sense of the term prior art. "Art" is an 18th century term for technology, *e.g.,* "useful arts" in the U.S. Constitution, and is a term still used to this day. We've encountered the phrase "person having ordinary skill in the art" (PHOSITA), meaning an

ordinary skilled but non-inventive engineer in a particular field (who, as a nice legal fiction, is presumed to know the entire prior art).

Prior art must of course be "prior," *i.e.*, before the priority date of the patent, which is typically when P applied for the patent (it might also be the date of an earlier related patent application). For our sample '398 facade-server patent, that date is Nov. 17, 2003. To try to show P's claim was anticipated, D needs prior art from before then.

Of course, a patent granted before that date is prior art. Patents are the primary source that patent offices use in researching novelty. Patents provide a structured format for a large chunk of prior art, designed to enable the types of searches that patent offices perform.

But anything that was reasonably "publicly accessible" before that date is also prior art. This could include patent applications which, while not yet granted, are often published 18 months after the application date. It could also include non-patent literature (NPL), such as articles in scholarly journals, trade publications, books, papers or slides presented to public conferences, even a single copy of a thesis stored in a foreign university library, so long as a reasonably-diligent researcher, in the relevant field of technology, could have found it through an index.

The requirement for prior art is public accessibility; not necessarily that the prior art *was* publicly accessed, merely that it could have been, by a PHOSITA using reasonable diligence, at the relevant time. For example, database contents, web pages, and in some cases even contents of an FTP site, can be prior art -- even if we could now show, using server logs, that no one did access it at the relevant time. Naturally, questions arise, for example, when material is behind a paywall (something is likely still public, even if one has to pay for it), or password-protected (likely not public, if only employees can get the password).

Prior art can be in any language, from anywhere in the world. The U.S. definition of prior art once had a "not invented here" component relating to non-printed prior practice outside the U.S., but since the America Invents Act (AIA) of 2011, a technological practice anywhere in the world, prior to the relevant date, can also constitute prior art.

While patents are generally only enforceable within a single country (see Part 1), the practice of looking to prior art from the entire world makes sense. The alternative of looking only to prior art within a single country would yield an insular, parochial definition of "invention." It would

make the first *importer* of something already known abroad into an "inventor." This would constitute a return to the mercantilist or protectionist origins of patent systems in 15th century Venice, which awarded patents for a "new ingenious contrivance, not made heretofore *in our dominion*."

This broad availability of prior art helps explain why a patent granted by the government could turn out to have been wrongly granted, *i.e.*, invalid. The PTO actively looks for prior art, and has thousands of employees trained to do so in numerous different fields (see the "Prior art already found by the PTO" section below), but a patent examiner is limited to about 20 hours (spread over several years) to decide on the validity of a single patent application.

Let's say a piece of prior art existed at the relevant time in some obscure, but still publicly-accessible, repository, somewhere in the world; the PTO happened not to find it and so granted the patent; P sued D for infringement of the patent; and now D -- financially motivated in a way that the PTO (which tends to view patent applicants as its "customers") was not -- finds the prior art. Publicly-accessible is not inconsistent with not-yet-found or difficult-to-find. The newly-uncovered prior art was, in essence, hiding in plain sight. Of course, D must show that what it found today was also accessible to the PHOSITA at a sufficiently early time to make it *prior* art. But you can see the possibility of P's seemingly rock-solid patent claim turning out, all along, to have been precariously perched on treacherous ground (see "Shaky property" below).

Generally, prior art must "enable" the PHOSITA, by teaching (possibly through the PHOSITA's ordinary, non-inventive, inferences, and some reasonable amount of experimentation) how to make and use the invention. A prior-art reference might have disclosed all limitations found in a later patent claim, but if the earlier reference did not also sufficiently disclose how to make and use the technology, it will be insufficient to invalidate the patent.

Most of the above discussion has assumed some sort of written material; we haven't said much about earlier non-written evidence of technological practice. Prior art need not be a patent, nor a "printed publication" (even in the very broad sense of that term), however, it can be a technology, component, practice, product, or service itself. A document from after the priority date may serve simply to establish the earlier existence of a non-documentary piece of prior art (see, *e.g.*, *In re Epstein* on "later-published abstracted descriptions of the software products;" for a similar issue from a completely different area of technology, see a fascinating European Patent

Office (EPO) case regarding native Kalahari prior practice, documented much later, and the hoodia appetite suppressant).

That actual real-world practice, apart from any formal publication, can constitute prior art, is important to software patents: Software *itself* (not merely documentation or other descriptions of the software), if publicly accessible, would be prior art relevant to software patents. Of course, prior art would include open source -- which would likely be viewed as a "printed publication" under the broad U.S. patent-law definition of that term. On the other hand, prior art would *not* include proprietary source code that is not publicly accessible. Trade secrets are not prior art: if some third party does X behind closed doors, X is not prior art that would invalidate P's later patent claim; in this respect, patent law rewards inventors who disclose, over inventors who try to keep it to themselves (of course: it's a "patent" system, not a latent system, nor a blanket reward for sheer ingenuity; it's largely about providing notice and disclosure).

The public-accessibility requirement for prior art is not arbitrary, but central to the purpose of the patent system, which aims to incentivize not just inventions as such, but *disclosure* of how to make and use them (another type of IP, trade secrets, can protect non-disclosed inventions). To slightly complicate this point, there is a category of "secret prior art" we won't get into further here: patent applications not yet published at the relevant time, and so visible only to the PTO and not to another patent applicant, but eventually later granted, are somewhat surprisingly prior art as of the earlier (secret) application date, not just the later publication or grant dates (the rationale for this was set forth in a 1926 US Supreme Court case).

Binary or object code distributed as a publicly-available product could be used as prior art, if it informed a PHOSITA (possibly using reverse engineering) how to make and use the relevant technology. Later we'll see that one of the problems the PTO faces with prior art for software patents is the lack of a full-text searchable database of software prior art (how, apart from what has been called "Big Code Mining," applied to commercial binary/object code, would the PTO examiners search the Windows, OSX or iOS operating systems themselves, as opposed to any publications regarding them, and as opposed to their source code, which is not publicly available?).

How a patent is compared with prior art

When comparing a patent claim to prior-art patents or published patent applications, the claim is compared, *not* with the prior-art claims as such (which only stake claims to inventions, without necessarily teaching how to make and use them), but rather with the *entire disclosure* of the prior-art patent or application.

So, to return of our running example of the ‘398 facade-server patent, we are looking for something from before November 17, 2003 teaching how to make and use the same thing as described in claim 1 of the facade-server patent. This “same thing” means disclosure (though probably not in a form that looks like a patent claim) of *each and every limitation* of the claim. This is an important point that is often overlooked. We can’t simply look for earlier examples of what seems like the patent’s core or gist (here what the patent calls a facade-server). We can’t scour through material from before November 2003, find a facade-server (likely under a different name), and proclaim victory, as would likely happen in a politicized online discussion of “bogus” patents.

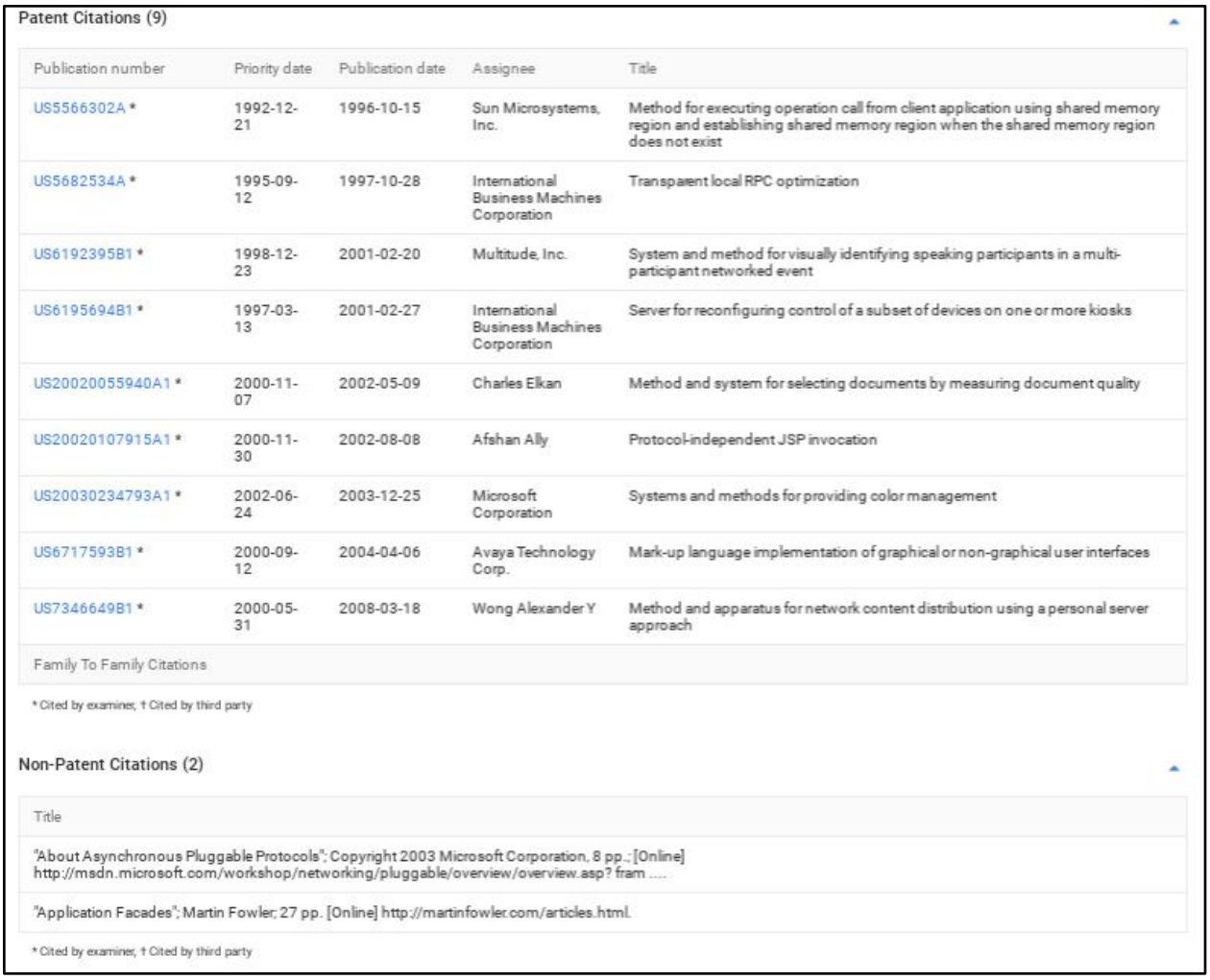
The point made earlier in this series regarding infringement, that the patent is really its claims, applies here too. We are *not* trying to show that some central concept or key idea of the patent was anticipated, *i.e.*, was not new. That might be true, but would in itself not invalidate the patent. A patent can be granted for a novel combination of previously-known ingredients or components. In fact, almost every patent claim includes at least some previously-known limitations, and one of those limitations might be something that superficially seems like the core of the patent (*e.g.*, what the owner put in the patent title). To knock out a patent claim as non-inventive, we must find each and every limitation, either combined in one prior-art reference (invalidity) or, as a fallback, in a combination of references (obviousness). Remember too, from Part 3, that what separates the claimed invention from the prior art may be a single small (though non-obvious) limitation added following initial rejection by the PTO examiner.

Prior art already found by the PTO; reading the file wrapper

Most patents cite one or more pieces of prior art. Why doesn't this prior art by itself keep the patent office from granting the patent? Because the patent applicant had successfully distinguished its claims from these (possibly very close) prior-art references.

Thus, to invalidate this already-granted patent claim, we'll likely need *different* prior art from what the PTO already saw. This gets back to the rebuttable presumption of validity discussed earlier. Given that the PTO has already granted a patent, despite having already seen certain prior art, overcoming the presumption of validity is going to be easier with something that the PTO didn't already bake into its determination of validity. It would be especially helpful for D to find a prior-art reference that the PTO didn't know about, but that P did know about, because as noted above, failure to disclose known prior art to the PTO can render the entire patent unenforceable.

Before seeking a pre-November 2003 facade-server, we should check the prior art already cited in the patent. Figure 1 below shows the cited prior art for the '398 patent, as reflected in Google Patents:



Patent Citations (9)

Publication number	Priority date	Publication date	Assignee	Title
US5566302A *	1992-12-21	1996-10-15	Sun Microsystems, Inc.	Method for executing operation call from client application using shared memory region and establishing shared memory region when the shared memory region does not exist
US5682534A *	1995-09-12	1997-10-28	International Business Machines Corporation	Transparent local RPC optimization
US6192395B1 *	1998-12-23	2001-02-20	Multitude, Inc.	System and method for visually identifying speaking participants in a multi-participant networked event
US6195694B1 *	1997-03-13	2001-02-27	International Business Machines Corporation	Server for reconfiguring control of a subset of devices on one or more kiosks
US20020055940A1 *	2000-11-07	2002-05-09	Charles Elkan	Method and system for selecting documents by measuring document quality
US20020107915A1 *	2000-11-30	2002-08-08	Afshan Ally	Protocol-independent JSP invocation
US20030234793A1 *	2002-06-24	2003-12-25	Microsoft Corporation	Systems and methods for providing color management
US6717593B1 *	2000-09-12	2004-04-06	Avaya Technology Corp.	Mark-up language implementation of graphical or non-graphical user interfaces
US7346649B1 *	2000-05-31	2008-03-18	Wong Alexander Y	Method and apparatus for network content distribution using a personal server approach
Family To Family Citations				

* Cited by examiner, † Cited by third party

Non-Patent Citations (2)

Title
"About Asynchronous Pluggable Protocols"; Copyright 2003 Microsoft Corporation, 8 pp.; [Online] http://msdn.microsoft.com/workshop/networking/pluggable/overview/overview.asp? fram ....
"Application Facades"; Martin Fowler; 27 pp. [Online] http://martinfowler.com/articles.html.

* Cited by examiner, † Cited by third party

Figure 1: Prior art considered by the PTO, in deciding to grant the '398 patent, as shown in Google Patents.

The PTO examiner and the patent applicant argued over several of these prior-art references, and these discussions appear in the patent's so-called "prosecution history" or "file wrapper," which we discussed in Part 3. The patent examiner used prior-art references to challenge the original applied-for claim 1, and this challenge motivated amendments to claim 1 that was granted (again, see Part 3). In particular:

- Patent 6,717,593 disclosed an "interactor" that can download XML and JavaScript via IPC (inter-process communications) instead of via an HTTP connection, when the server is co-located on the same computer with the interactor;

- Published application 2002/0107915 disclosed a web server using an "adapter" to service non-HTTP requests, with a Java Server Pages (JSP) file executed for non-HTTP requestors.

These both sound similar to how the facade-server hosts a local application and makes the local app's I/O accessible to a web browser. Yet we know the patent was still granted, after back-and-forth discussion between owner HP and the PTO, of such earlier disclosures. As seen in Part 3, the patent applicant distinguished claim 1 from these prior-art disclosures by noting that the facade-server does not have a network connection; it also added as claim limitations a "program" that creates an interface between the facade-server and the web browser; and "without opening network ports."

Any prior art we find now will have to do better than those already cited in the patent. While D could argue that the PTO mistakenly accepted P's asserted distinctions between these already-known references and P's claim, and mistakenly accepted P's amended claim limitations as sufficient narrowing of the claim to avoid the prior art, that is unlikely to be convincing.

True, the PTO has only limited time to examine each patent application (as noted earlier, about 20 hours spread over several years), has an inadequate database of software prior art (I've discussed this in a section on "The Problem of Undocumented Prior Art" in an article on using reverse engineering to uncover software prior art), and is not as motivated as an infringement defendant to find invalidity, but on the other hand, the patent examiner or supervisor has likely worked with many patents in this same area. Here, the application that became '398 was handled by "Art Unit" #2194, which specializes in inter-process communications (IPC; yes, there are thousands of government employees -- about 8,000 patent examiners -- specializing in areas from resins, peptides or proteins, to tobacco, to resilient tires, to IPC, VMs, compression, language translation, and the like). This particular examiner had dealt with many similar patents before this one was granted. For anything he's already put his eyes on, we're unlikely to do better than he in using it to invalidate the patent.

So, we'll want something the examiner hasn't seen. Further, additional references for a local-enabled web server would be merely "cumulative" of what the PTO already knew when it granted the patent.

Since the PTO presumably does a reasonable job of searching its own patents, we should remember to mine non-patent literature (NPL). While the PTO will use NPL, including the internet (see, *e.g.*, MPEP 904.02(c)), as seen from Figure 1 above, neither non-patent citation came from the examiner.

Searching for a negative limitation in prior art

To find prior art for the facade-server limitation of the software patent claim we're trying to invalidate, what we need are not web servers that provide local access, but ones that do *not* provide network access; not a server that provides local access, but one that *only* provides local access.

A pinpointed search for this would require searching for an explicit negative. A prior-art reference that merely talked about local access, without mentioning network access, would likely be insufficient to anticipate the "without" negative limitation. We need something that explicitly teaches "local only" or "the web without a network connection." On the other hand, the limitation only eschews network access in relation to the facade-server. The web browser for example can still make non-local calls. However, *any* web server's CGI hosting is likely to be local, so the server we're looking for must be purely local (and so by extension will reside on the same computer as the web browser).

While it is difficult to encode these negative requirements into a pinpointed search, keywords and synonyms similar to those we used in our infringement search (Part 3) are likely to be useful here too. Recall that P has accused D's product X of infringement; D will want its search to include prior art that is as similar as possible to X: that which (P says) infringes, if after, must -- even if D doesn't agree it would infringe -- necessarily (by P's own stance) invalidate, if before.

Google Patents enables searches (in patent and scholarly materials) for keywords and synonyms within a date range. Given a particular patent number, the keywords Google picks for its "Find Prior Art" feature appear superficial -- here, it chose (based on the patent as a whole, rather than a particular claim) web, application, server, data, means -- but these are easily changed to something like: local, cgi, server, browser, cli (command-line interface; our brainstorming in Part 4 of infringement search terms included cli AND browser).

A prior-art example

As one example, this Google search yielded a paper, "Integrating a Command Shell Into a Web Browser," presented at Usenix in June 2000, *i.e.*, over three years before the '398 patent's application date. The paper discusses a prototype web browser named LAPIS, for which binary and source code is also accessible on the web (at least once one knows the "lapis" search string), dated October 2003. Recall our priority date is in November 2003; one frequently encounters such close calls in patent litigation; perhaps it reflects an "inventive minds think alike" or "in the air" phenomenon.

If we were seeking prior art simply for the web browser as a front-end to legacy apps, the title alone suggests we struck gold. But remember, we need a "facade-*server*." It is surprisingly common for patent litigants to point to one side of a pair (such as client/host, send/receive, or browser/server) as though it were evidence for the other side of the pair, with a barely-articulated assumption: "well, we found a receiver, so there must be a sender." (Or worse, "there's a sender so there must be a receiver.") Even so, using this prior-art browser to match a server claim element might be possible if we gave it some thought.

Remember that the "facade-server" we're looking for in the prior art is unlikely to come neatly labeled as such. The patent owner coined this term. For better or worse, there is no predefined set of the parts to which patent claims must refer. There is a patent classification system with granularity at the level of VxDs, RAID metadrivers, and SCSI device drivers for example, but this serves a different purpose than helping to write claims. On the contrary, the patent owner can be its "own lexicographer."

So, if such a thing as a facade-server is in the prior art, it's likely labeled as some other kind of server -- or perhaps not labeled as a server at all. A server by any other name would smell as sweet. This however raises some questions which will strike some readers as interesting teasing out of edge cases, and others as who-cares "semantics":

- If something, whatever it is called, is hosting apps, and delivers content to a web browser, is it thereby a web server?

- What if one part of the browser makes the "request" via a simple function call to a different part of the browser? Is the callee thereby a server (especially here where "server" by P's own definition in the file wrapper is "software only")?
- Does message-passing invocation, within a single app, make it any more reasonable to call it a "server" than would API function invocation?

These are the sorts of questions which have to be answered whenever someone asserts that X, while perhaps not the most typical exemplar of Y, nonetheless still fits, if barely so, within the category Y. In part because P and/or D often stretches the use of a term to include edge cases, or points to examples at the very edge of that term's meaning, semantics often rule in patent claim construction during patent litigation. Engineers often scoff at this as "lawyer games," but it's not very different from a software engineer checking whether a variable or parameter can take on a certain boundary-condition value.

With the above questions in mind, consider the following excerpts from the "Integrating a Command Shell Into a Web Browser" paper about a web browser acting as a command shell (emphasis added):

- "HTML interfaces for local programs. Currently, programs with HTML interfaces must be installed in a web server in order to handle form submissions. LAPIS can submit forms to local programs by the Common Gateway Interface (CGI), an existing standard used by web servers. This opens the possibility of running HTML applications entirely locally. HTML offers benefits of both a graphical user interface (GUI) and a command-line interface (CLI).... As a demonstration, we have wrapped an HTML interface around the Unix find program."
- "CGI scripts can be run directly by the [LAPIS] browser-shell, without installing them in a web server"
- "Although CGI is commonly used by web servers to invoke external programs, no major web browser can invoke a CGI program locally. (The closest we've found is the Help View in KDE 1.1, which displays HTML help documents and uses CGI to invoke a local search engine.)"

- "For security reasons, LAPIS only executes a cmd: URL if it originates locally -- e.g. if it is typed into Location box or found in a page loaded from the local filesystem. A link in a remote web page cannot invoke a Tcl command"

In other words, this browser provided its own built-in server-like functionality (CGI) for running local-only apps.

Can we legitimately say that the LAPIS browser -- by supporting CGI solely for local apps accessed via local use -- was thereby acting as a facade-server, *i.e.*, as a web server limited to local access? And even if so, can we say that this paper would in 2003 have *enabled* (taught the PHOSITA how to make and use, without "undue experimentation") claim 1 of the facade-server patent?

Limitation-by-limitation analysis of a prior-art reference

Unless we summarily dismiss the browser-acting-as-server assertion as nonsense, the right way to answer these questions is to run through each limitation of claim 1, and compare it to the LAPIS paper. (To keep things simple, I'll treat the LAPIS paper, and the LAPIS software itself, as a single reference that can be analyzed for anticipation. But while a diligent reader of the paper would have been motivated at the time to download the software, that might still be a combination of multiple references; see the Obviousness section below).

Let's once again haul out our sample patent claim, so we can carefully check for each limitation's presence in the prior-art reference:

- [1pre] 1. A computer system comprising:
    - [1a] a central processing unit (CPU);
    - [1b] a memory unit coupled to the CPU;
    - [1c] an application stored in the memory unit and executable by the CPU;
    - [1d] a facade-server stored in the memory unit and executable by the CPU; and
    - [1e] a program stored in the memory unit and executable by the CPU,
    - [1f] wherein the program creates an interface between the facade-server and a web-browser for exchanging data associated with the application,

- [1g] wherein the facade-server hosts the application without utilizing network protocols and without opening network ports.

[1a] **CPU** and [1b] **memory**: The LAPIS browser is a Java JAR file which loads into memory; its instructions are executed by the Java virtual machine (VM), which at the time was well-known to emulate a CPU, so we'll treat these as "gimmes."

[1c] **Application**: As quoted above, the LAPIS paper states, "we have wrapped an HTML interface around the Unix find program." Installing and testing the LAPIS browser on Windows (the binary software is itself prior art, even if one didn't have the Usenix paper or source code) showed it was also able to run the Windows tasklist program and grep.exe (see Figure 2 below); find, tasklist, and grep are all applications.

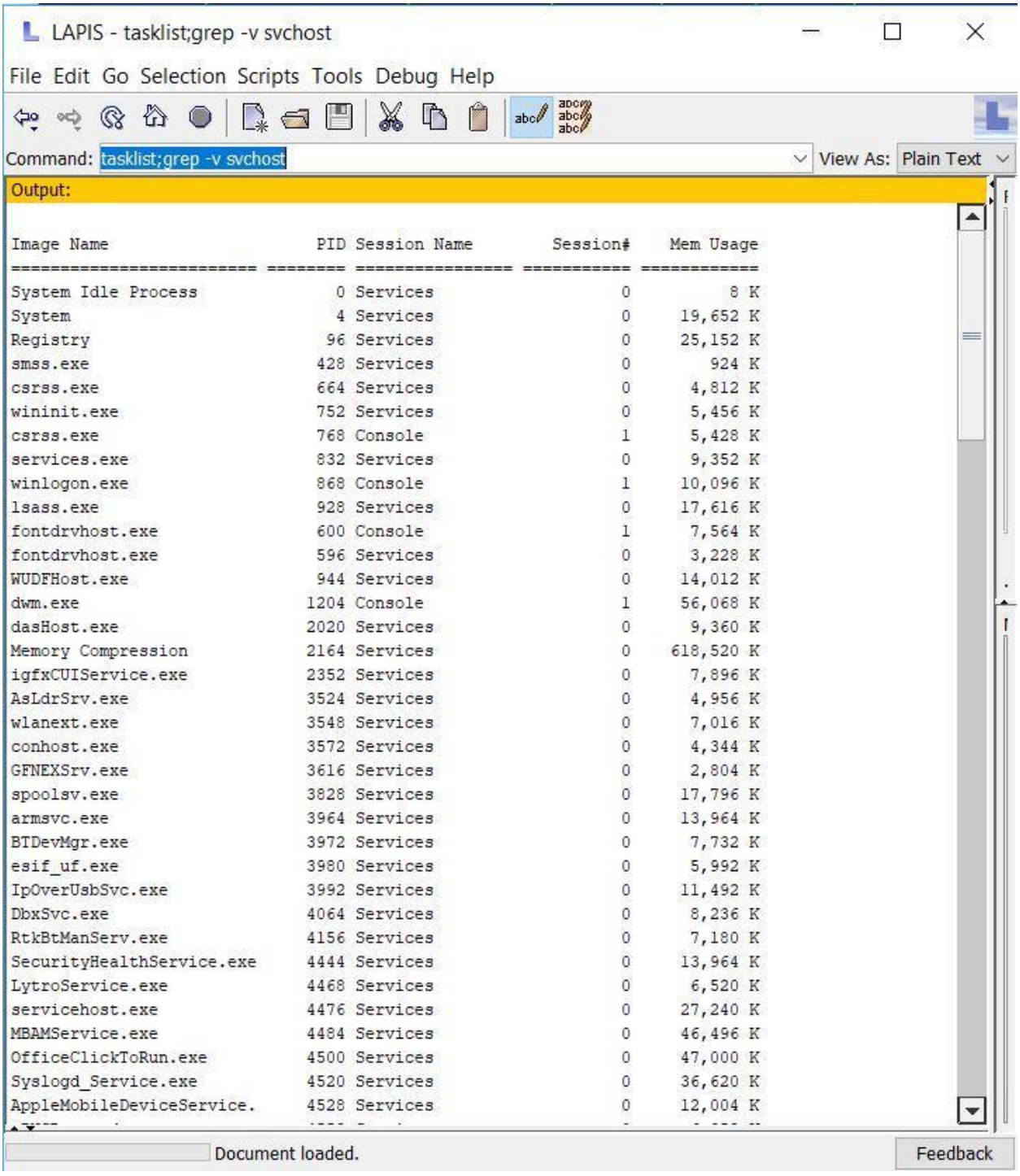


Figure 2: The LAPIS command-shell web-browser from 2000-2003, running the Windows tasklist.exe command, and feeding its output to the grep.exe program, to filter out any lines containing "svchost". LAPIS automatically took output from a command, and fed it to the next command.

[1d] **Facade-server**: While limitation [1d] itself doesn't say so, to be a facade-server per claim 1, something must satisfy several requirements (see the discussion of "constraints" in Part 4): it

must (a) be a web server (albeit one without a network connection), per the '398 spec and file wrapper; (b) host the application without utilizing network protocols and without opening network ports, per limitation [1g]; and (c) reside on the other side of the data-exchange interface (created by the "program") from the web browser, per limitation [1f]. Let's deal with each separately:

(a) Web server: For LAPIS to be invalidating prior art, the facade-server must be that part of LAPIS that provides the CGI interface used to host and run apps such as find, tasklist, and cmd. Without getting into details of the LAPIS source code, two identifiable Java classes in LAPIS are responsible for this; we can point to different classes, albeit with a single JAR file, as distinct components of an apparatus. But doesn't a "server" need to be open to requests from multiple clients? To rephrase our earlier questions: If part of a browser gets content by issuing a function call to a different part of the browser, is the callee thereby a server? Here, the browser also displays non-local web pages, which might contain attempted access to local commands, which are deliberately blocked: as quoted above, "For security reasons, LAPIS only executes a cmd: URL if it originates locally." Thus, LAPIS presents a network-capable browser, but with a built-in command server that is purely local.

P has a good response: the paper itself, as quoted above, says CGI scripts can be run by the "browser-shell, without installing them in a web server." That language strongly implies that the "browser-shell" is *not* a "web server." D might respond that P is making too much of this language, which could mean "without installing them in a [separate] web server."

(b) Host app without network protocols or ports: CGI invocation happens within the browser, without HTTP even to localhost. The browser makes an exec() call. As quoted above, the browser excludes attempts at non-local access to local commands.

(c) Other side of interface from web browser: The interface between the nominal server and browser portions of LAPIS is Java class method invocation. As a possible alternative (litigants can "argue in the alternative"), the LAPIS paper refers to "creating pipelines of web services and local programs," such as using the local find command to extract data from a web page. D might refer to such pipelines as a relevant interface, though the reference to "web services" might interfere with the no-network constraint (note how interactions among claim limitations create constraints on how each limitation is mapped to a product or to prior art; any infringement or

invalidity assertion is a constrained mapping of the limitations that comprise a patent claim on the one hand, with the attributes/features/parts of an accused product or prior-art reference on the other hand).

Program (creates interface between facade-server and web browser): The program that creates the interface is LAPIS itself. If D needed the interface-creator to be a component distinguishable from other components, it could point to specific Java classes, rather than "LAPIS itself" as a monolithic block. For example, the LAPIS paper refers to two custom protocols, "cmd:" and "exec:", and D could point specifically to the location in the source code where these are handled.

Web browser: This seems simple enough; it's LAPIS itself. The browser uses HTTP to access resources on the web. This seems fine since the patent claim's no-network requirement applies only to the facade-server. But if we're pointing to the browser as also providing the facade-server, the no-network assertion is difficult. While the LAPIS paper states that its CGI is local-only, we still probably need an *identifiable portion of* LAPIS, which resides on the other side of the interface from the part of LAPIS we're calling the facade-server. Again, rather than point to "LAPIS" as a whole, we can point to specific Java classes, distinct from those that use CGI to host apps, that provide browser functionality and that reside on the other side of the interface.

As a general point, patent litigants should try to think in terms of granular components, which are better suited to pinpointing infringement or invalidity than are larger assemblies. (Of course, litigants would prefer to leave things at a more general level, to give themselves some wiggle room: the phrase "for example, but not limited to," appears frequently in infringement contentions).

To use this "Integrating a Command Shell Into a Web Browser" paper as prior art for the facade-server claim, D would likely need an expert to plausibly assert that, while referring to a "browser," the paper nonetheless informed the PHOSITA at the time of a local-only CGI server, which is a facade-server, contained within a browser; experts turn their experience and knowledge into something that can be cited as a piece of evidence: it is sometimes said, in a non-pejorative way, that "experts create facts."

P will perhaps have an expert who responds that, for something to be a server, albeit a local-only one, it still must service general requests via some interface, and that function calls within a single Java app do not count as interface requests; though if P's expert wouldn't say the same for *message*-passing invocation with a single app, this sounds overly formalistic.

Thus, using this paper as prior art raises claim-construction questions for "server" and "interface." These are similar to the earlier claim-construction issue for infringement (see Part 1), in which D would likely argue that localhost-only HTTP still counts as a "network protocol."

P may further argue that, even if the paper did describe an instance of what P would later call a facade-server, this required D's expert's later inference, and the earlier paper would not *at the time* have taught or enabled the PHOSITA to make and use a facade-server. Enablement must have been contemporaneous with the patent claim's priority date. Experts sometimes forget this, and opine in present-day terms, rather than speaking to what the PHOSITA at the relevant time would have understood.

Whether or not D decides to assert the LAPIS paper and software as prior art, it should also pay attention to the paper's remark, quoted above, about KDE 1.1 desktop help search using local CGI. This is likely relevant prior art. D should also think about other old local-web technology, such as Microsoft's one-time support for so-called HTML Applications (HTAs), though these are likely to raise the same "okay, so where's the server?" question as the LAPIS paper.

P should argue that D's finessing of the "server inside a browser" issue, by pointing to distinct individual Java classes within the LAPIS browser's Java archive file, required that D pull in a second prior-art reference (the LAPIS software) that is distinct from the conference paper describing LAPIS. As noted above, showing invalidity by anticipation requires finding a single prior-art reference disclosing each and every limitation of a claim.

D might meet this challenge by asserting that the conference paper, which contained a link to the software, thereby "incorporated by reference." More important, D can point to the LAPIS software itself as a single prior-art reference, with the conference paper, binary code, and source code, as mere evidence for what LAPIS would have disclosed to the PHOSITA. Remember that publicly-available software itself, with or without any descriptions of the software, can be prior art, once someone locates it.

The reader may be understandably annoyed that we are taking so long, in 2019, to discuss an antiquated technology like local CGI. To reiterate a point made earlier in this series, we truly don't care about this particular technology, or this particular patent claim (which was picked more or less at random from a list of short patent claims). The arbitrary choice of this patent claim means that the annoying little issues we encountered (such as whether a particular software module can be considered a "server"; whether using HTTP solely for localhost still constitutes using a "network protocol"; and so on) are fairly typical of what we would have spent time on, had we chosen a claim from a different software patent.

Obviousness: "motivation to combine"

What if D had to concede that the conference paper and related software are two separate prior-art references? Or what if D has a single prior-art reference, but one or two limitations were missing or only barely, arguably present, but readily found in some other prior-art reference? Can D combine them to try to invalidate P's patent claim? Whether a prior-art *combination* renders a patent claim invalid depends on an obviousness analysis. Obviousness is a large topic within patent law, which we'll only briefly discuss here, with more detail at another time.

While obviousness is sometimes called "the very heart of the patent system," the "crowning glory of patent law," and the like, a defendant seeking to invalidate a patent would prefer to do so with anticipation rather than by having to work through what we'll see is a more-complex obviousness analysis (at the same time, D shouldn't overly stretch an anticipation argument, merely to avoid obviousness). This is similar to a point made earlier in this series that while equivalence for P is a second-best fallback from literal infringement, equivalence nonetheless receives far more attention in patent-law treatises. Equivalence and obviousness are perhaps seen as rendering patent law more principled or more subtle than the more-common literal infringement and anticipation.

Obviousness, like equivalence, is a term of art with a specific meaning in patent law, just as, for example, "stack" and "queue" have specific meanings in software engineering. One can't blithely declare that "that patent claim is obvious" or "that part of the product is not equivalent to this limitation of this patent claim," without first going through the legal tests for obviousness (see below) or equivalence (see Part 4).

To show that P's patent claim, while not anticipated by the prior art, was nonetheless an obvious extension of the prior art, such that non-inventive practitioners could have done it, D generally takes two (or rarely, more than two) prior-art references and shows that the PHOSITA at the time would have been motivated to combine them to yield the claimed invention. Generally, most limitations are found in one reference, with a missing limitation or two supplied from the second reference.

D will assert, for example, "Claim 1 is invalid under 35 USC 103 on being obvious over [prior-art #1, *e.g.*, the LAPIS paper] in view of [prior-art #2, *e.g.*, the LAPIS source code]."

The prior art showing obviousness cannot come from anywhere, but must come from the same field as the invention, from the "analogous arts," or from the nature of the problem to be solved (see, *e.g.*, clamshell case). The gap between the primary prior-art reference and the patent claim may also be explained by pointing to the common knowledge of the PHOSITA at the relevant time (likely as testified to by D's expert), rather than to a specific publication (see the Supreme Court's KSR decision: "In many fields there may be little discussion of obvious techniques or combinations").

Obviousness is analyzed via a four-step process, which determines:

1. The scope and content of the relevant prior art;
2. Differences between the prior art and the claims at issue;
3. The level of ordinary skill in the pertinent art; and
4. Objective indications of non-obviousness ("secondary considerations"), such as commercial success, long-felt but unsolved needs, and failure of others to make the claimed invention.

In essence, one determines if the difference (#2 above) between the patent claim on the one hand, and the relevant prior art on the other, was within the level of ordinary (non-inventive) skill (#3). This is not so much an arithmetic comparison (is #2 < #3, *i.e.*, is the difference of claim minus prior art less than PHOSITA skill?), as a decision whether something in the prior art and/or general knowledge could have *pointed* the non-inventive PHOSITA to combine the multiple prior-art references to yield the claimed invention.

This contemporaneous pointing at the future patent claim is called "motivation to combine" (MTC), which in turn is based on what (as a result of the *KSR* case) has become a "flexible" teaching/suggestion/motivation (TSM+) test: is there something in the prior art that could have given the PHOSITA a reasonable expectation of success in combining different bits of the prior art to solve a particular problem or yield a certain result?

D will assert something like, "One of ordinary skill in the art would have been motivated to combine [prior-art #1, *e.g.*, the LAPIS paper] with [prior-art #2, the LAPIS source code], or alternatively to combine [prior-art #1] with [prior-art #3, *e.g.*, KDE 1.1 help search using CGI], in order to solve the problem of using a web browser as a display engine for local commands, because...".

The temporal aspect to obviousness is sometimes overlooked: the issue is not whether a claim is now obvious, but whether it can be shown to have been obvious (within the skill level of a non-inventive engineer) in some past time, generally when P applied for the patent. Much of the complexity of obviousness is aimed at avoiding hindsight (which is also why we said in Part 3 that the idea of using a patent claim as a "scavenger hunt checklist," while a useful metaphor for part of an infringement investigation, is *verboten* in obviousness analysis: any combination of multiple limitations from disparate sources could be made in retrospect to look "obvious" -- so patent law tries to prevent this).

Part of the obviousness test is negative, asking whether something in the prior art would have "taught away" from the combination reflected in the future patent claim. Such teaching away of course would help show the claim was non-obvious. Here, the "conventional wisdom" ca. 2003 might have discouraged the PHOSITA from considering a web server without network protocols or ports, even in the face of references showing a web server on localhost, and even in the face of existing firewalls that could be configured to prevent any outside access.

As noted above, one part of obviousness analysis includes consideration of "objective" indications of non-obviousness, such as commercial success, long-felt but unsolved needs, and failure of others to make the claimed invention. This is a mandatory part of the analysis. For example, if P had embodied its patented invention into a commercial product, and this product had sold like hotcakes (with consumer demand driven by a feature that depended on the facade-server, rather than by, *e.g.*, a celebrity endorsement), P could use this as some evidence of non-

obviousness. Likewise, if others before the time of P's patent application had tried, without success, to employ a web browser as a secure display engine for character-mode apps.

Shaky property: the risk of patent invalidity

The preceding discussion of patent claim invalidation, by showing anticipation or obviousness, has been from the perspective of defendant D, defending itself from plaintiff patent-owner P's assertion that D infringes a patent claim belonging to P. D in response had asserted that it not only doesn't infringe, but that P's patent claims are invalid in the first place.

As noted earlier, merely by bringing suit against D, patent owner P is motivating D to try to invalidate P's patent. This is an example of the tensions or balance in IP law that we've touched on in this series: On the one hand, it's been made fairly easy for P to sue for infringement (P is allowed to end up being wrong about infringement, without necessarily paying D's attorney fees, as long as its case was plausible, and P need only show D's infringement with a preponderance of the evidence). But on the other hand, bringing this suit increases the risk of P effectively losing its patent: "Use it and [possibly] lose it." We've also noted that, if P's patent is invalidated in P v. D1, it's likely going to remain invalid against any subsequent D2, D3, etc. In effect, P must win on validity each and every time.

Patent-infringement litigation is not the only time patent validity is challenged. Of course, as we saw above, the claim would have originally been challenged by the PTO in the years between when P originally applied for the patent, and when it was granted (here, 2003-2008). We saw in Part 3 that P amended its claim during this lengthy process between P and the PTO.

Even after a patent is granted, P or "any" third party may request a re-examination based on patents or publications it brings to the PTO's attention. P itself might want to do this in order to strengthen its patent against possible later challenge: "What doesn't kill me makes me stronger."

The America Invents Act (AIA) in 2011-2012 created a new process, called *inter-partes* review (IPR) which further allows patent validity to be challenged (including by defendant D in ongoing patent litigation) before a newly-created Patent Trial and Appeals Board (PTAB). Because the PTAB is at the very government agency (the PTO) that granted the patent in the first place, should the PTAB decide to hear an invalidity challenge (it often declines), there is *not* the

presumption of validity we've discussed. A government agency is merely reconsidering its earlier decision, as the Supreme Court noted in its *Oil States* decision (though isn't there some difference between taking away an existing government grant, albeit one wrongly given, and not granting it in the first place?).

I said earlier that one can see the possibility of P's seemingly rock-solid patent claim turning out, all along, to have been precariously perched on treacherous ground. But is this risk of patent invalidity ever-present? Even some strong patent advocates believe so; for example, IPWatchdog (which provides patent-attorney training and sells DIY provisional-patent-application software, and thus one would think has a vested interest in promoting the value of patents) states that inability to "quiet title" on patents is "the single most damning problem facing the patent system today" and that the US patent has become an "illusory promise" and "an illusory right that only fools owners into hoping and believing they have some property right."

Without necessarily agreeing with this strong language, or with the view that this is qualitatively different from the pre-AIA status of patents, or for that matter from property in general (which as noted in Part 1 defies any naive absolutist view of immutability), at any rate patents are a shaky form of property. Perhaps it is unrealistic to expect too much stability and predictability in a field which, after all, has to do with invention, *i.e.*, "something new under the sun."

While some invalidity will later be seen to have been apparent all along, from within the four corners of the patent itself (remember, the PTO has only about 20 hours to examine a patent application, and so will sometimes get it wrong), or from the prior art already considered by the examiner (see above), the ability of later-discovered prior art to invalidate a patent claim offers a particularly dramatic ending to P's property. Of course, the later-uncovered evidence must still be shown to have earlier been publicly accessible. But given the (necessarily) broad definition of public accessibility, P may understandably feel as if its suddenly-collapsed home was all along sitting on an unknown (yet knowable) geological fault.

There are some important counterbalances to the shaky nature of patents. For example, if D asserts a particular prior-art challenge in an IPR and fails to show invalidity, or even if D could have made the assertion but didn't, D cannot make that assertion (it is estopped) in later or ongoing litigation. In this sense, P's survival of an IPR makes it stronger.

In addition, it is always possible for P to amend its claim to preserve or regain validity, albeit to a smaller territory, as Amazon did, years after the original patent grant, with some of its once-invalidated "one-click" patent claims.

Indeed, P can even amend already-granted patent claims for the very purpose of covering D's activities occurring after the grant of P's patent; this resembles the so-called "Texas sharpshooter" phenomenon in which one shoots at the side of a barn, and only after draws bullseyes around the bullet holes. Importantly, the original patent specification must provide support for the newer amended claims, which in turn leads in part to very lengthy patent specifications (the idea that ultimately "you get what you disclose, not what you claim," discussed in Part 2).

As a parting thought, given that there is title insurance for real property, wouldn't one expect analogous patent validity insurance? Of course, there is insurance for potential defendants being sued for patent infringement, and there is patent-litigation funding available for potential plaintiffs, but apparently not patent-validity insurance as such. One patent enforcement insurance policy "reimburses your costs due to any invalidity counter lawsuits filed against your IP," but this only covers costs, though it does also include an "IP Quality Review" in which the insurer may suggest reexamination or reissue to strengthen a possibly weak patent prior to litigation.

Conclusion

In this Part 5, we've covered the following topics:

- The concept of patent invalidity: patent claims might have been wrongly granted, and subsequently found to be invalid.
- Patent claims can be invalidated for any of the reasons that would keep a patent from being granted in the first place: anticipation (lack of novelty), obviousness, lack of enablement, and so on.
- Showing anticipation or obviousness depends on the prior art, which consists not only of earlier patents, but also of published patent applications, and printed materials; in fact, prior art is anything publicly accessible (for obviousness, from the relevant field), from anywhere in world, in any language, and may even include non-written public activity or

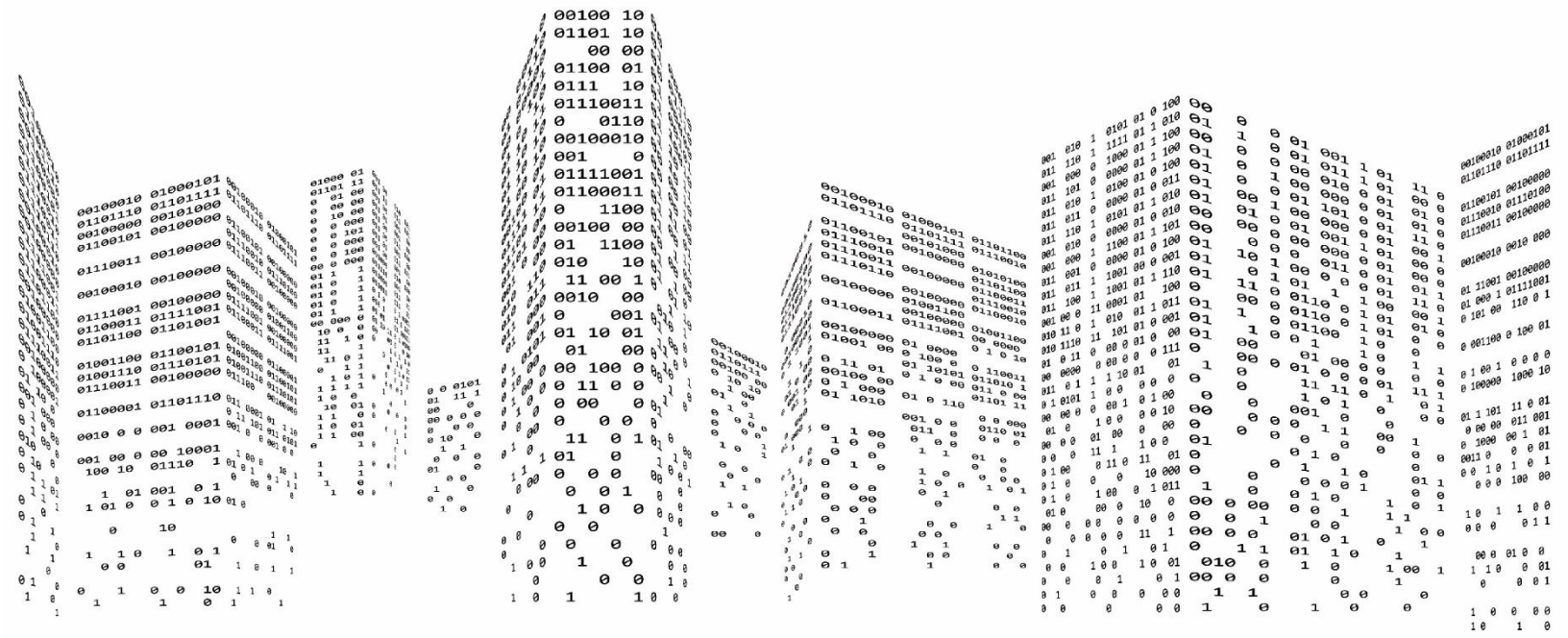
sales. It can include software itself, in addition to written descriptions of, or source code for, the software.

- While prior art must have been publicly accessible in some sense, it might be in an obscure location, and so can pop up unexpectedly, given a sufficiently-motivated prior-art search.
- Patent invalidity, like patent infringement, is shown on a claim-by-claim, and limitation-by-limitation basis, and is not a matter of some core concept of the patent.
- Some claims of a patent may be invalid while others remain valid.
- Showing patent invalidity and non-infringement, like showing patent validity and infringement, often involves "semantic" arguments over edge cases.
- If invalidity cannot be shown by pointing to a single prior-art reference that anticipated each and every limitation of a patent claim, one can fall back on obviousness, by showing that an ordinary non-inventive practitioner in the field (the PHOSITA) would have been motivated to combine multiple prior-art references and/or common knowledge in the field, to yield the (thereby-obvious) "invention."

The next and final Part 6 will discuss:

- Claim charts for analyzing infringement and invalidity;
- Local Patent Rules (LPR) in key federal districts, focusing on court requirements for infringement and invalidity contentions and responses;
- The need for a reasonable pre-filing investigation and "plausible" infringement contentions when bringing a patent infringement case;
- Discovery: getting information the other side has; and
- Very briefly touching on the many issues we've managed to skip over, despite the length of this introduction to patent litigation, including: venue and "forum shopping"; the Markman claim construction hearing; summary judgment; settlement or trial; remedies (injunctions and monetary damages); willful infringement and attorney's fees; design-around; importation, "domestic industry," and the ITC; and appealing adverse rulings to the Court of Appeals for the Federal Circuit (CAFC).

Andrew Schulman is a Senior Software Litigation Consultant at DisputeSoft. He focuses on software patent litigation, pre-litigation investigations, and source-code review. Mr. Schulman is also the founder and principal of Software Litigation Consulting. As a software engineer, he edited and co-authored several books on the internal operation of Microsoft operating systems, and is an attorney with an LL.M. in Intellectual Property.



If you are in need of a software patent dispute expert, we invite you to consider DisputeSoft.

Contact Information

Jeff Parmet, Managing Partner

301.251.6182

jparmet@disputesoft.com

12505 Park Potomac Ave. | Suite 475 | Potomac, MD | 20854