Text2PTO:
Modernizing Patent Application Filing

A Proposal for Submitting Text Applications
to the USPTO

The Big Picture

- The USPTO is building a new, modern patent examination software system.
- One part of this is modernizing how patent applications are filed.
- Applicants will receive many benefits from this with minimal changes in their existing work practices.

Main Goal

Provide a safe, easy way for applicants to file the **text** version of their patent applications online.

ments. These and other changes can be made to the platform in light of the above detailed description.

[0048] In general, in the following claims, the terms used should not be construed to limit the platform to the specific embodiments disclosed in the specification and the claims, but should be construed to include all systems and methods that operate under the claims. Accordingly, the platform is not limited by the disclosure, but instead the scope of the platform system and method is to be determined entirely by the claims.

[0049] While certain aspects of the platform are presented below in certain claim forms, the inventors contemplate the various aspects of the platform in any number of claim forms. For example, while only one aspect of the platform is recited as embodied in machine-readable medium, other aspects may likewise be embodied in machine-readable medium. Accordingly, the inventors reserve the right to add additional claims after filing the application to pursue such additional claim forms for other aspects of the platform system and method.

Currently: A Scanned Image

What is claimed is:

1. A multiple operating system method comprising:

initiating boot of a host operating system (OS) on a host central processing unit (CPU);

after initiating boot of the host OS, accessing non-host boot code for a first non-host OS that is stored in an areas of memory used for booting the host OS; and

initiating boot of the first non-host OS on a first non-host CPU from the host OS, wherein initiating boot of the non-host OS comprises an inter-processor-interrupt (IPI) sent from the host CPU to the first non-host CPU.

comprise at least one host partition, and at least one specialized partition for performing a specialized function.

7. The method of claim 1, further comprising:

installing a driver on the host operating system; and

configuring a platform using the driver, wherein configuring comprises partitioning the platform to include multiple partitions, wherein a partition comprises an operating system and at least one CPU allocated to the operating system, and wherein configuring further comprises determining particular platform resources to which each partition has access.

8. The method of claim 1, wherein the first non-host CPU is configurable to disregard broadcast inter-processor-interrupts (IPIs) sent by the host CPU.

9. The method of claim 1, wherein the first non-host operating system is a real-time operating system (RTOS).

10. A system comprising:

at least one host operating system (OS) operating on a physical processor (host CPU);

at least one non-host OS operating on a non-host CPU; and

a driver configured to operate under the host OS, to store a non-host OS boot code in an area of physical memory associated with host OS boot activities, and to direct the non-host OS to access the non-host boot code.

11. The system of claim 10, wherein the driver is further configured to configure a platform that includes the host OS and the non-host OS, wherein configuring comprises allocating platform resources to the non-host OS, the resources comprising CPUs and memory.

Related Applications

This application claims priority from U.S. application Ser. No. 11/242,378, filed Oct. 3, 2005, which claims priority from U.S. application Ser. No. 10/865,865, filed Jun. 14, 2004, which claims priority from U.S. Provisional Application No. 60/478,220, filed on Jun. 13, 2003.

This application claims priority from U.S. application Ser. No. 11/407,733, filed on Apr. 20, 2006, which claims priority from U.S. Provisional Application Ser. No. 60/763,577, filed on Jan. 31, 2006, and from U.S. application Ser. No. 10/857,009, filed Jun. 1, 2004, which claims priority from U.S. Provisional Application Ser. No. 60/474,551, filed on May 30, 2003.

Each of these applications is incorporated herein by reference.

Field of the Invention

The present invention relates to a method of coordinating and stabilizing the delivery of stored energy, such as wind generated energy stored in the form of compressed air energy.

Goal: Text

Background of the Invention

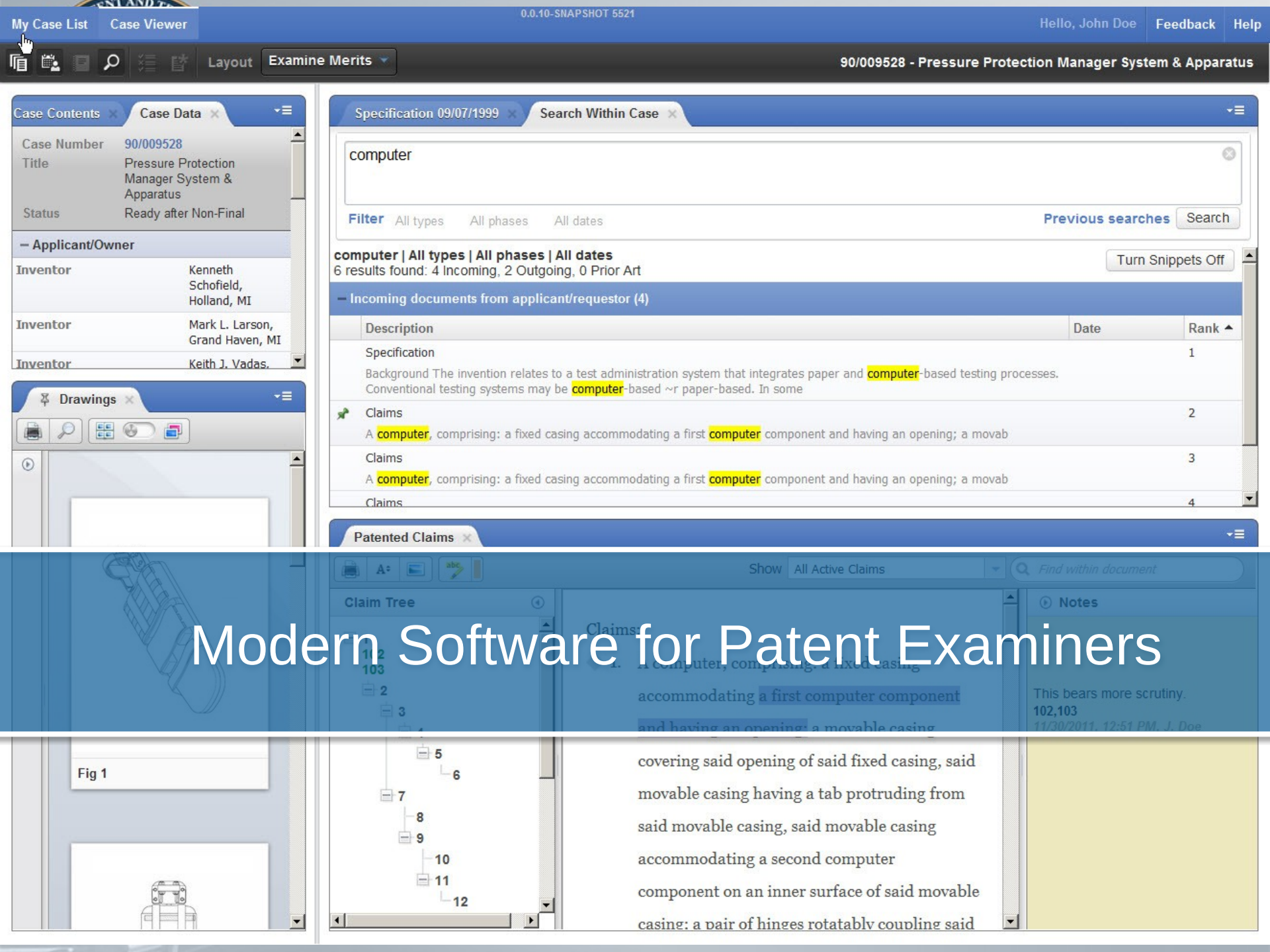
Generation of energy from natural sources, such as sun and wind, has been an important objective in this country over the last several decades. Attempts to reduce reliance on oil, such as from foreign sources, have become an important national issue. Energy experts fear that some of these resources, including oil, gas and coal, may someday run out. Because of these concerns, many projects have been initiated in an attempt to harness energy derived from what are called natural "alternative" sources.

Wind farms are best operated when wind conditions are relatively constant and predictable. Such conditions enable a consistent and

Benefits

- Increased accuracy in filings
- Automated formalities checks
- Additional useful automated information
  - Claims trees
  - Patent family information
- Will enable better tools for patent examiners
- Will provide large cost savings for USPTO

Modern Software for Patent Examiners



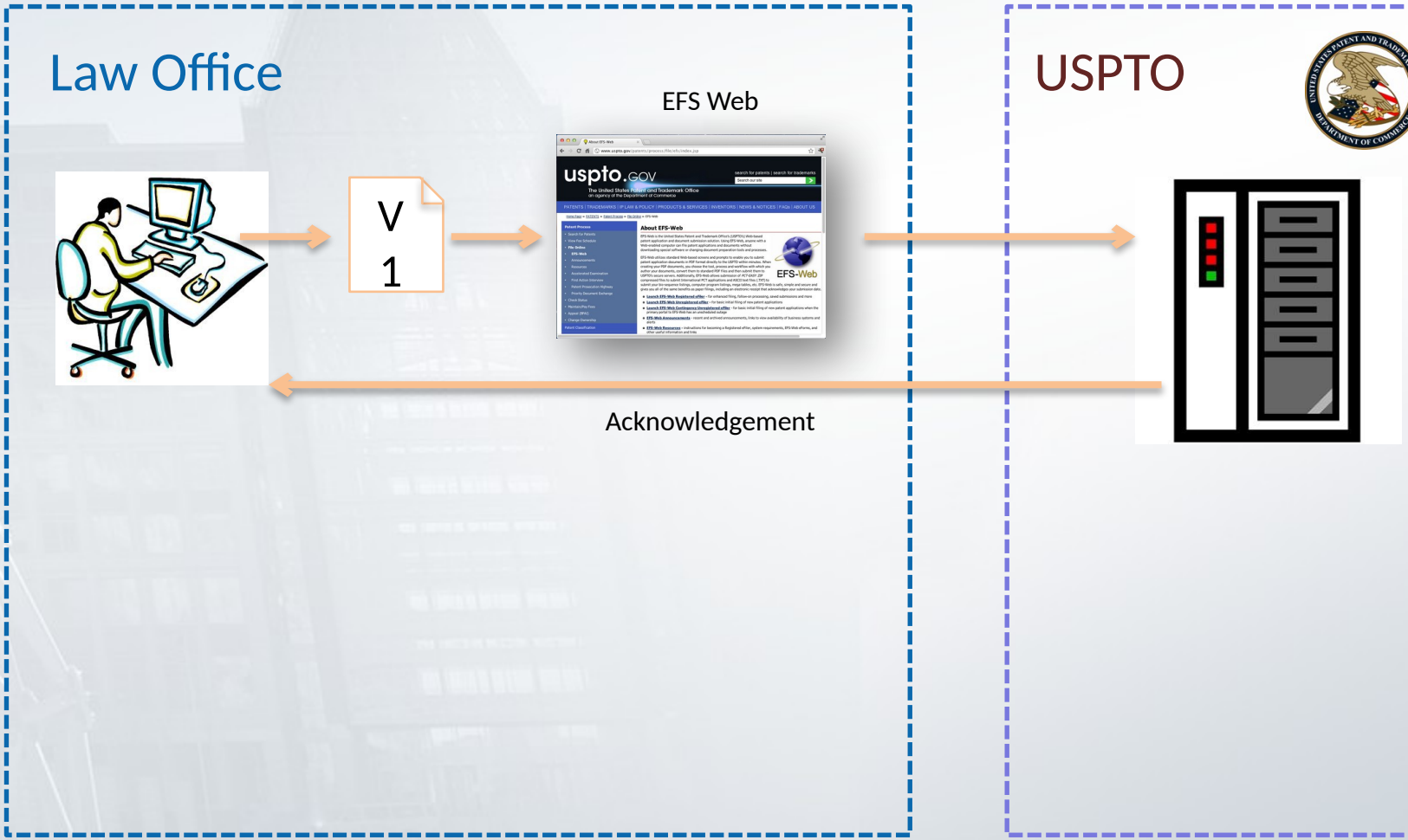
Focus:
Ease of Use for Applicants

- Eliminates the need to:
  - Scan or pdf the file
  - Upload four different files
  - Figure out the number of pages for each file
- Amendments can be made on a copy of the original.
  - Cover sheet can be conveniently added for you.
  - Amendments can be authored using Track Changes
- No software to download or install

MAIN IDEA

Author your application in Word, and then upload it t.

Author your application in Word, and then upload it to USPTO.



Applicants' Concerns

- What if we left track changes on?
- What if we left private information in?
- How can we be sure the contents of the application have not changed?

We Have Solutions for Applicants' Concerns

- What if we left track changes on?
  - USPTO is providing a validation tool that will alert you about this.
- What if we left private information in?
  - The validation tool will tell you this as well.
- How can we be sure the contents of the application have not changed?
  - USPTO is providing a code that allows you to easily and quickly verify that we have a verbatim copy of the text that you submitted.

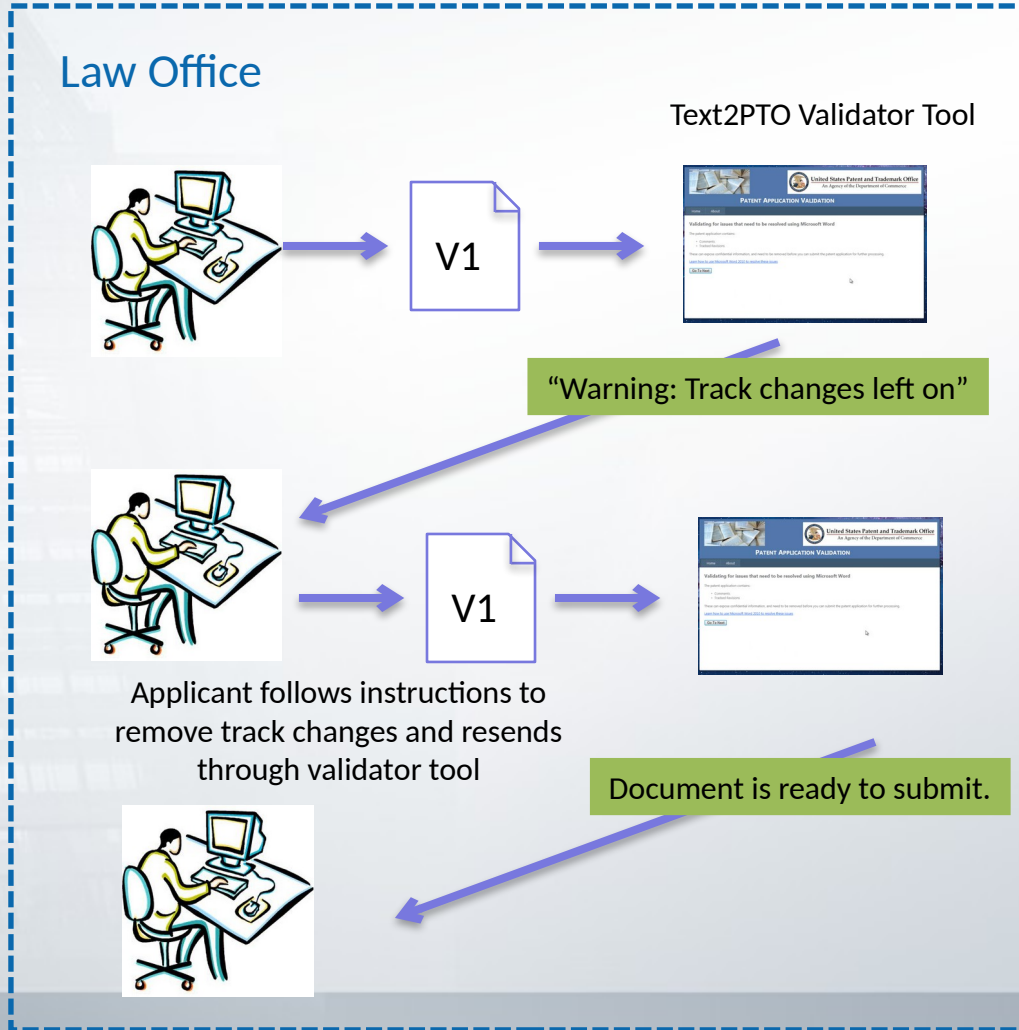
VALIDATOR TOOL, PART 1

Check for Track Changes and Private Information

Make sure you did not include any private information.

Run the validator tool in a web browser on your own computer.

It tells you if you included private information or left on track changes and tells you how to remove it.



What Video Should Show

- 2 minutes max (all timing info below is maximum)
- No need for audio; we'll speak over it.
  - Scroll through two different applications [45 sec]
    - Show the same application but with two radically different looks (fonts, layout, etc)
    - In one of the files, show there are some track changes visible but only if you switch the View (from final to final showing markup, for example)
  - First check:
    - Upload the document with markup into the tool [5 sec]
    - Show the warning message and the "how to" screen. [30 sec]
  - Show the user doing the repair, maybe doing "accept all changes"
  - Show the user re-running the tool on the file and getting an "ok" message.

VALIDATOR TOOL, PART 2

Check for Disallowed Markup (such as embedded fonts, colored text).

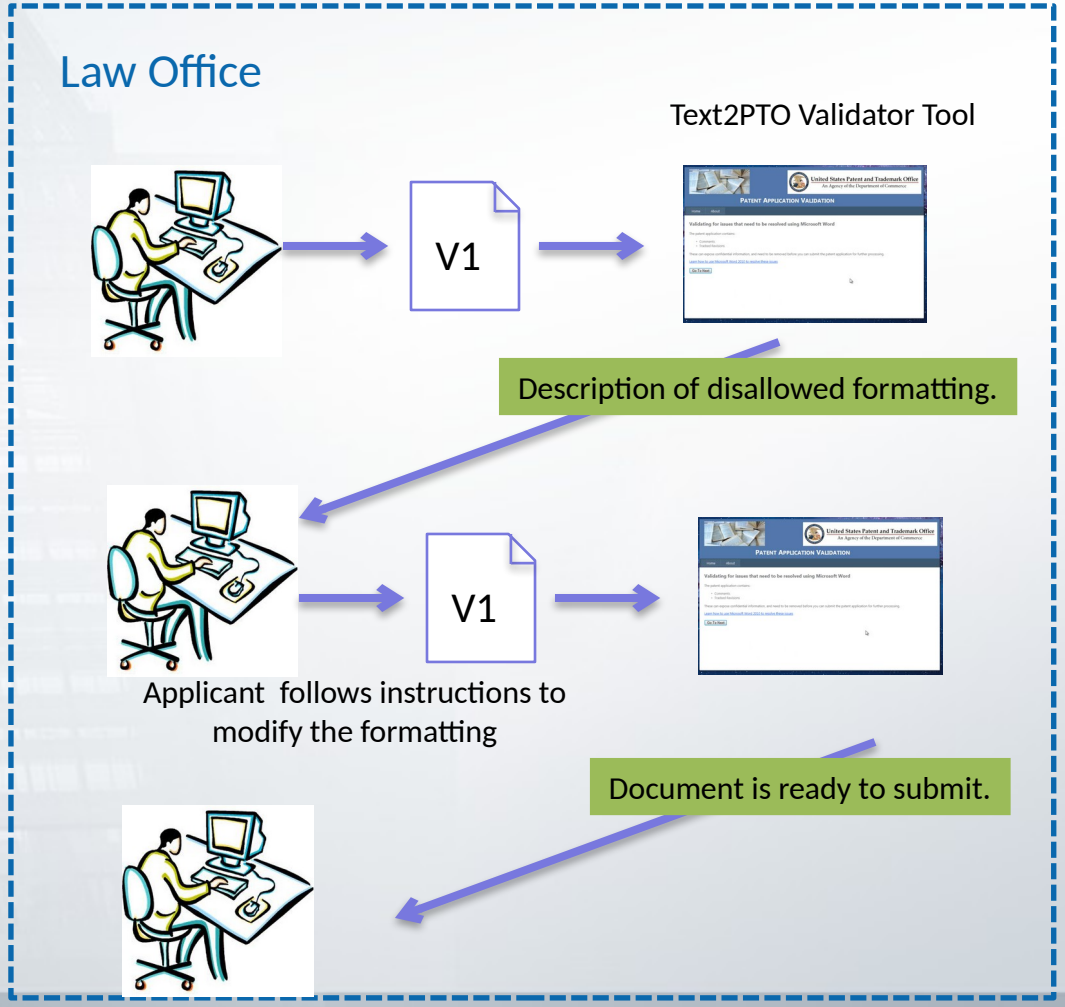
Optionally Fix It.

Check for and change any disallowed formatting.

Run the validator tool in a web browser on your own computer.

If your file has any disallowed formatting, either:

- Follow the instructions to remove it, or
- Let the tool remove it for you, step by step.



What Video Should Show

- Two examples
- 1st example: Text with color highlighting behind it. [30 sec]
  - Step through the tool detecting it and then step through the user correcting each one by one
- 2nd example: disallowed font [30 sec]
  - Show the “before” look
  - Show the user selecting an offered alternative and applying the change to all instances.
- Show the document now getting an “ok”
- Show the tool keeping track of and saving the latest version of the corrected file.
  - (Should they rename the file? Does the tool have access to a directory of versions?)

VALIDATOR TOOL, PART 3

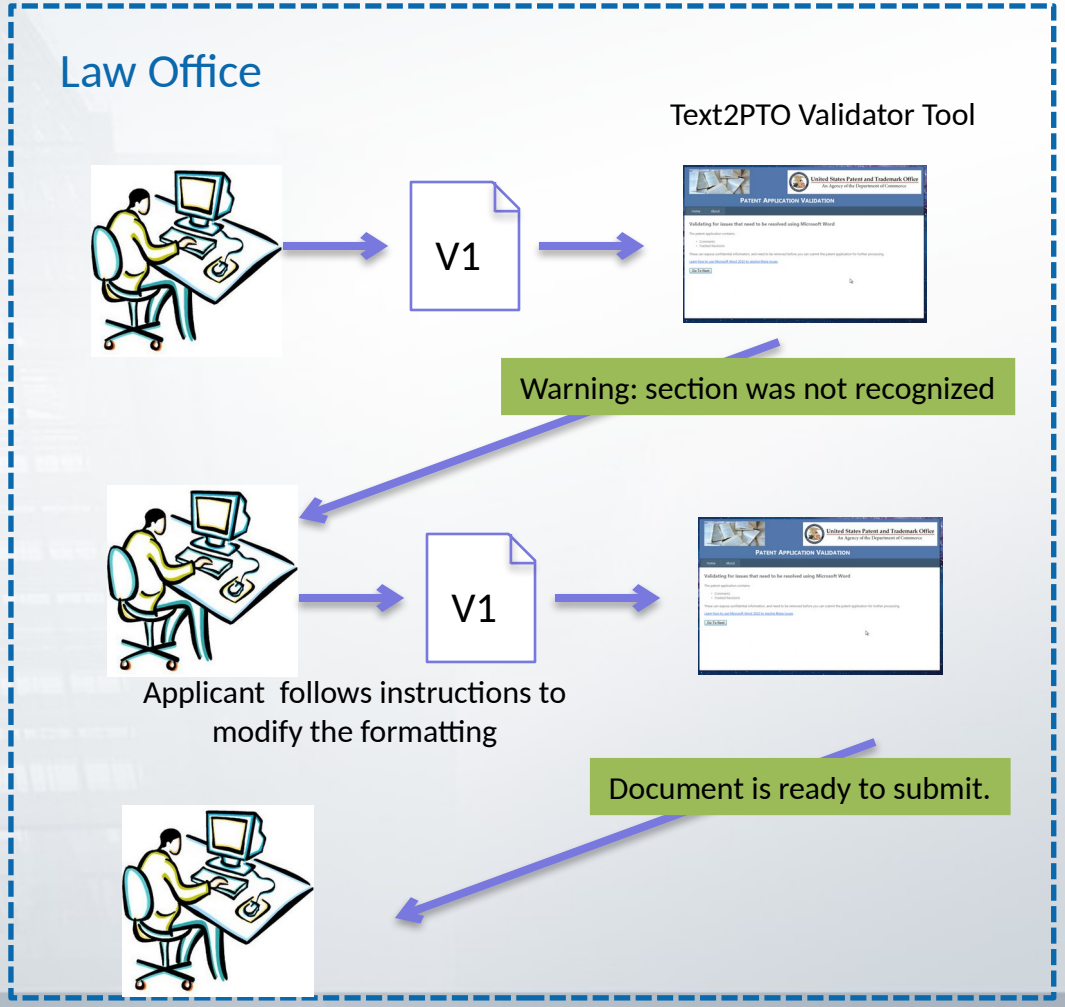
Make sure the section headings are each on their own line.

Make sure sections headings are recognized.

Run the validator tool in a web browser on your own computer.

Verify the tool found the abstract, claims, and specification sections.

Make small adjustments if not.



What Video Should Show

- Show two examples:
  1. Wrong words to identify the specification [30 sec]
     - Just have system say it can't identify the spec and then show the allowed wordings for this header.
     - Show the user editing the document and re-running the tool and getting an "ok" message
  2. Section heading not on its own line [30 sec]
     - Show system guessing the right location but warning user they have to put in a blank line.
     - Show user doing this and then getting an "ok" message.

ENSURING THE DOCUMENT REMAINS UNCHANGED

Text2PTO supplies an identifier is created based on the text itself.

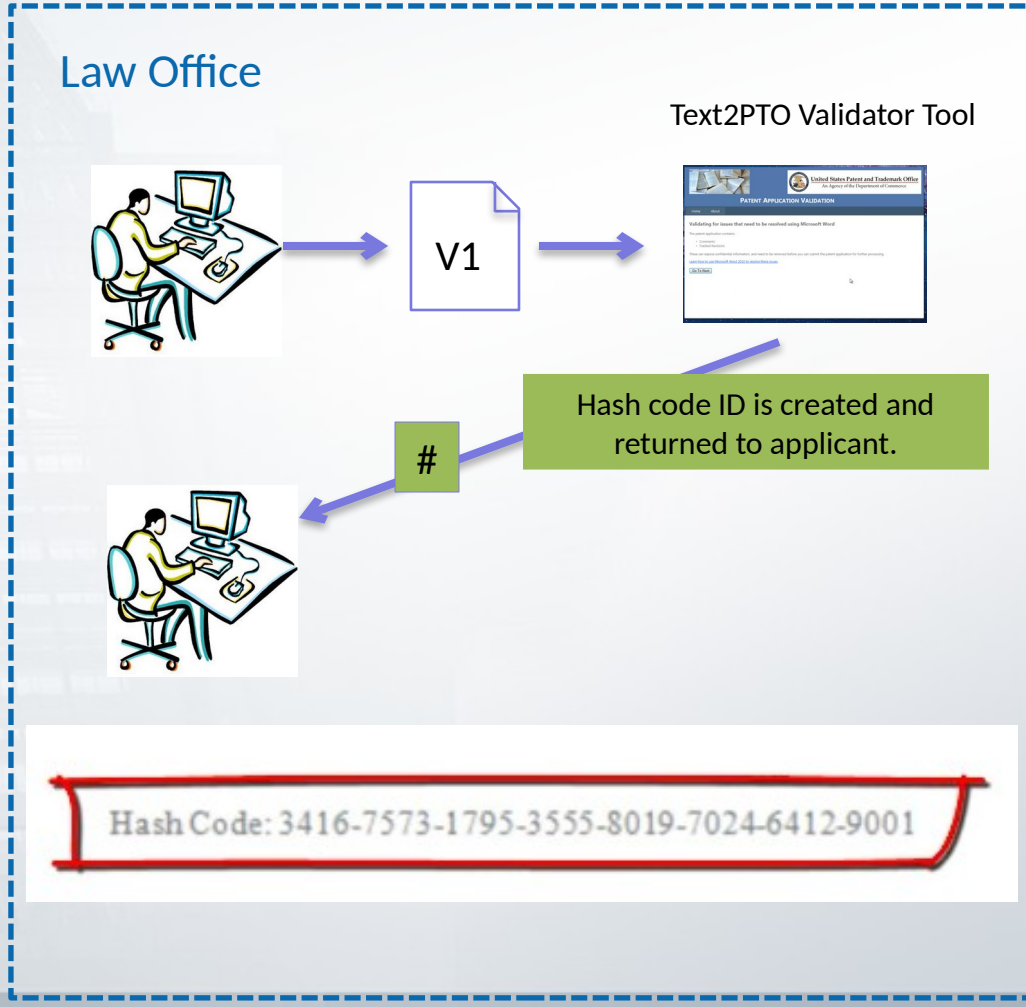
Ensuring the Document Remains Unchanged

- A standard way to do this is to create a unique identifier that is based on the actual text of the file.
  - This is called a “hash code”.
- The validator tool will produce this unique ID for you.
- If your document changes even by one space, the unique ID cannot be recomputed from the file.
  - This ID lets you prove to yourself and others that the file has not changed (or has changed if that is the case).

Create a unique ID for this version of the document.

Run the validator tool in a web browser on your own computer.

The tool generates a Hash Code ID and optionally places it on the document.



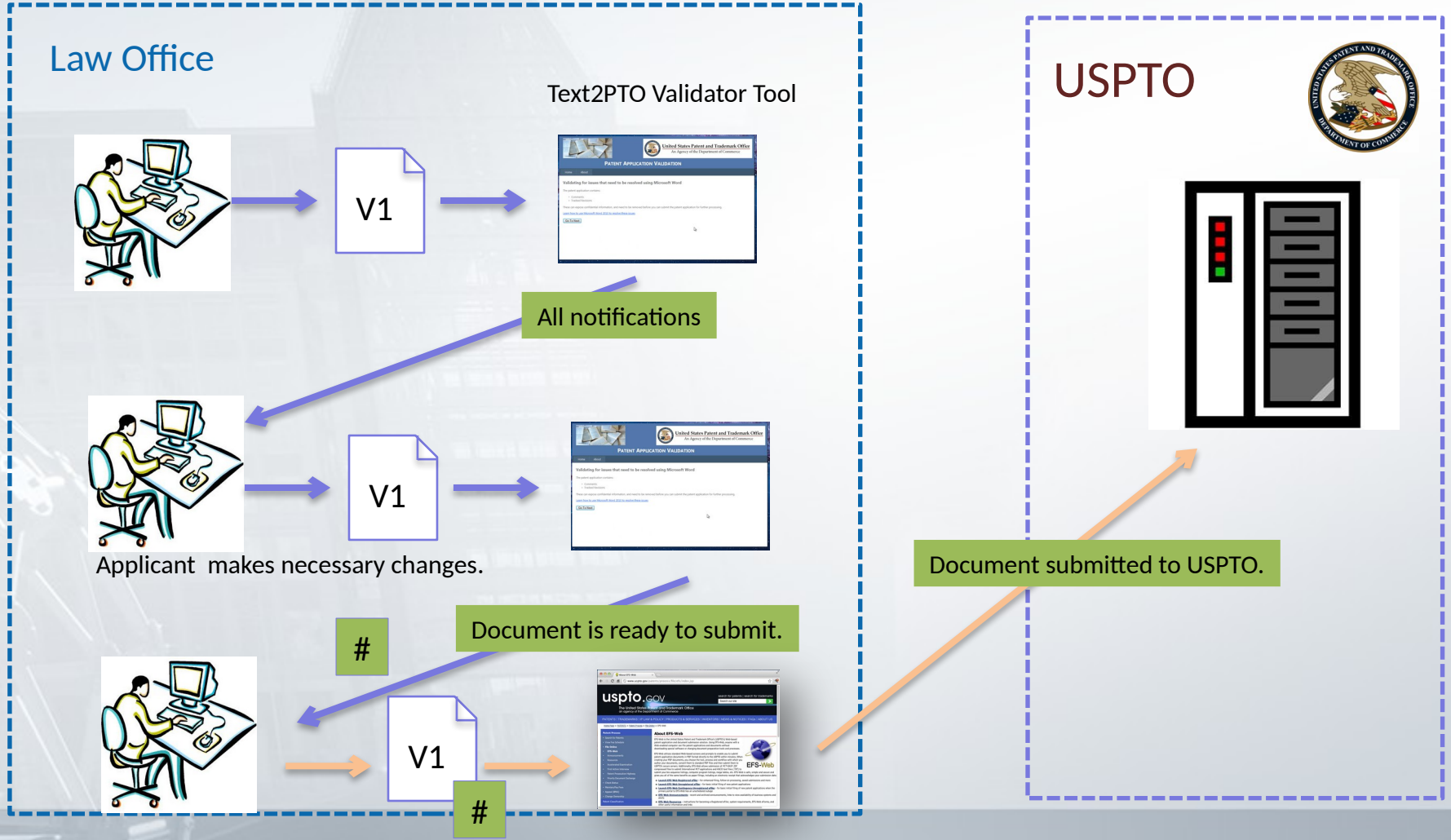
What Video Should Show

- 30 sec for this sequence
- Show user getting an “ok” message for the entire document
- Show message offering to create a hash ID from the document text.
- Show user accepting this.
- Show the resulting document including th hash tag.

How Long Does This Take?

- Only a couple of minutes if your document is properly formatted.
- If it isn’t, then we predict from 5 to 15 minutes depending on the number of changes needed.

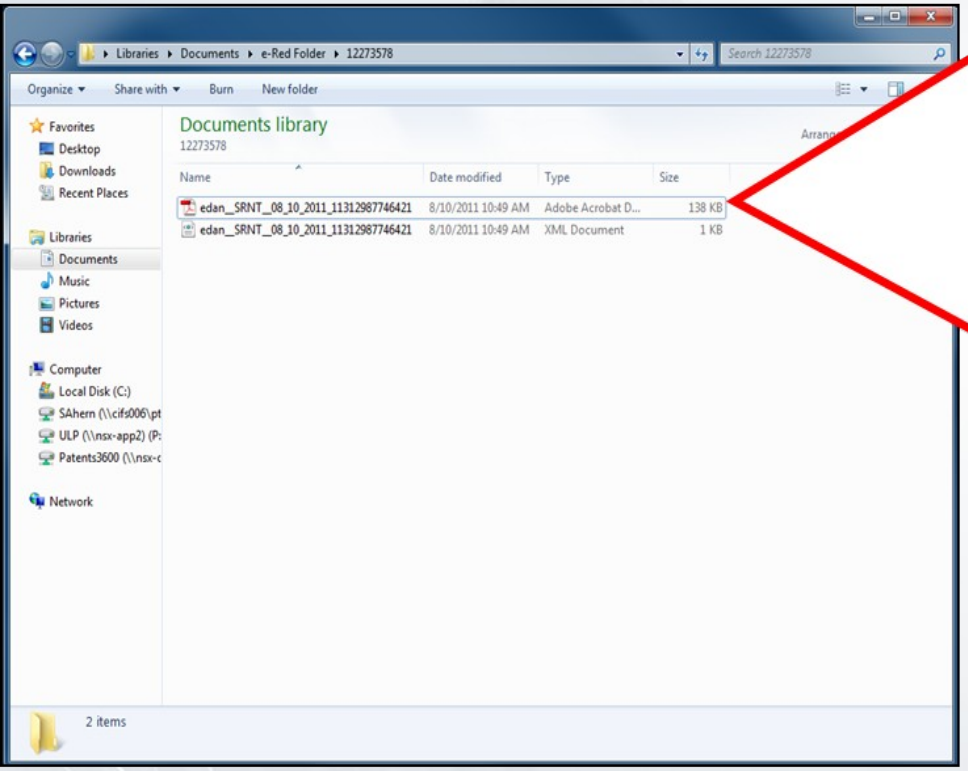
Make sure sections headings are recognized.

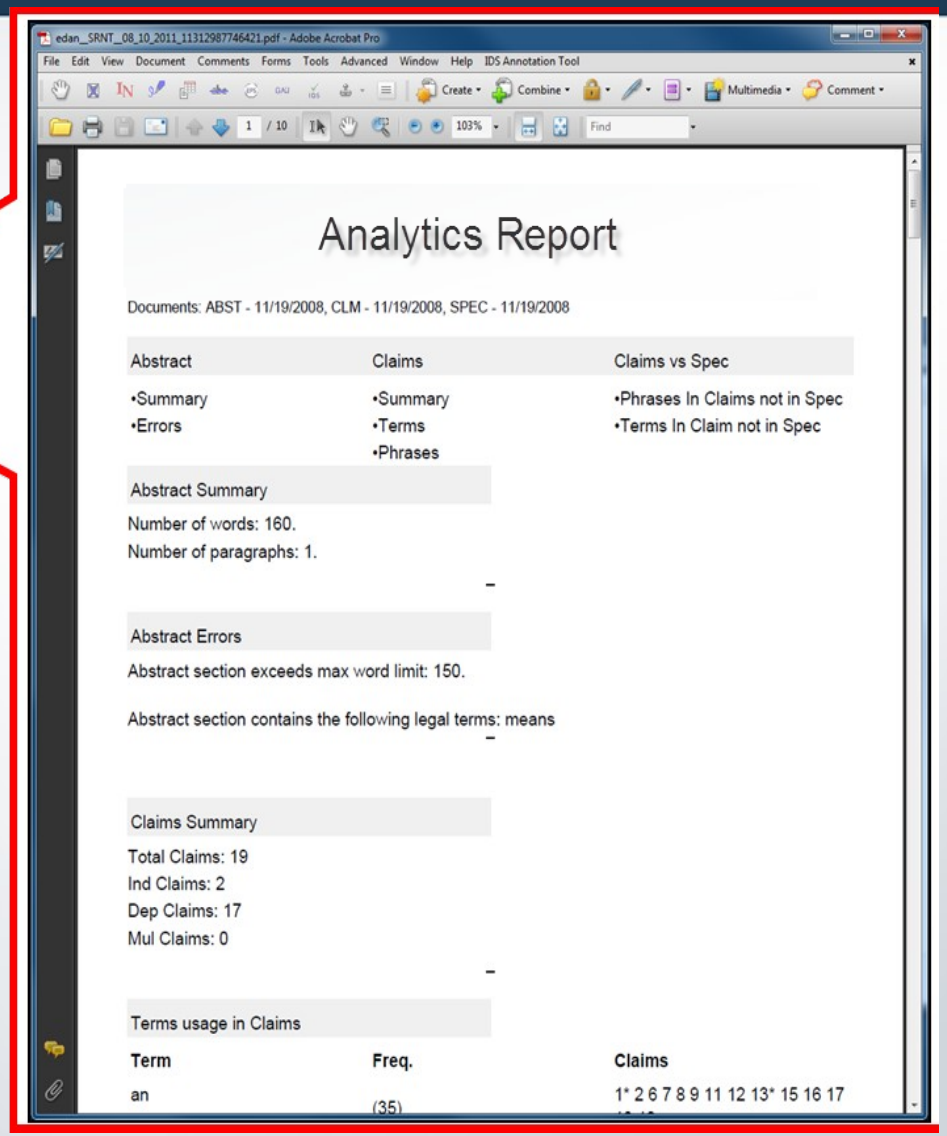


OPTIONAL:
FORMALITIES CHECKS

Check abstract length, show claims tree and any errors in claim numbering, see patent family information.

Results of Formalities Checks





Results of Formalities Checks (con't)

Analytical Report

Abstract	Claims	Claims vs Spec
• Summary	• Summary • Terms • Phrases	• Phrases In Claims not in Spec • Terms In Claim not in Spec

Abstract Summary

Reviews for Compliance with MPEP rules

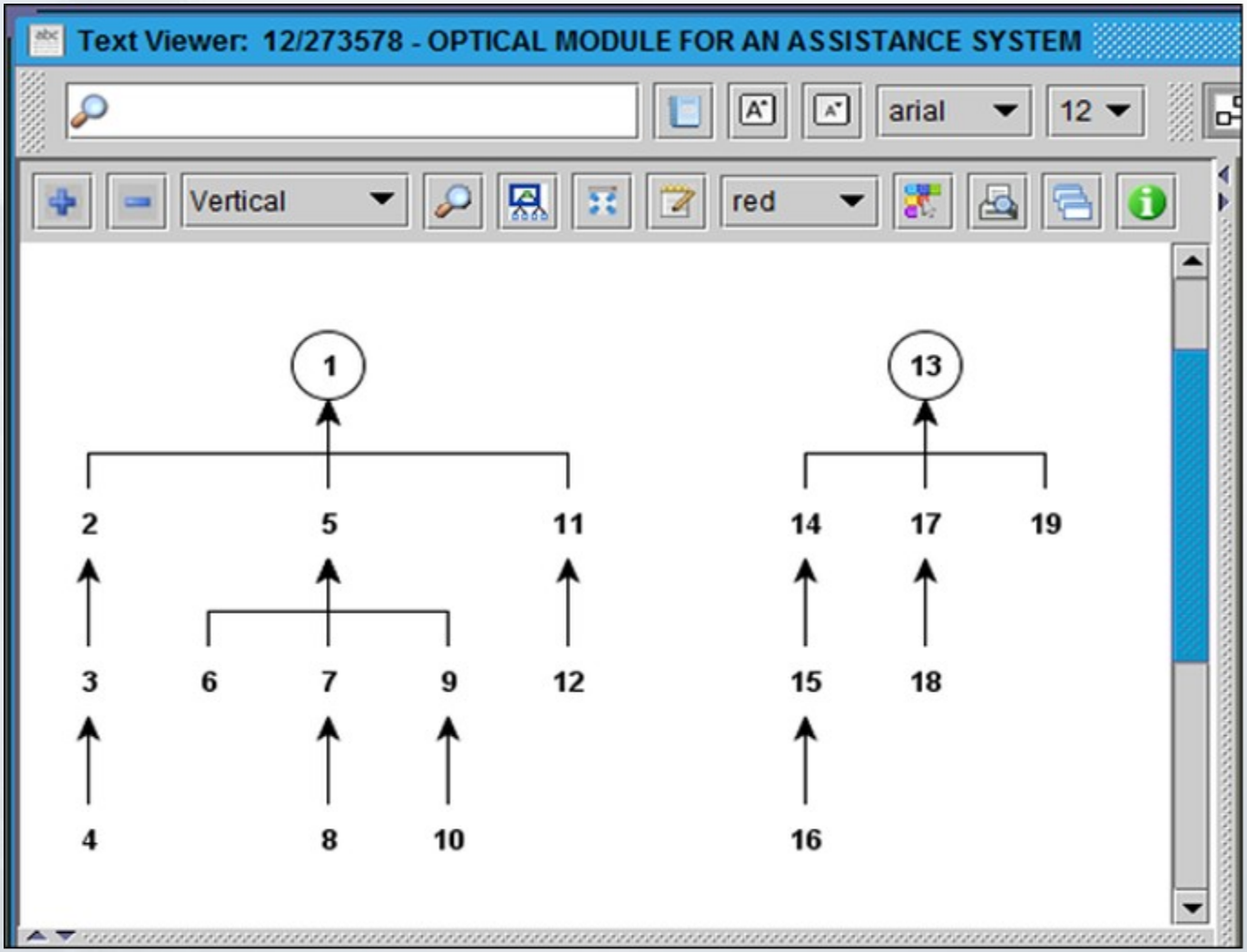
Number of words: 160.
Number of paragraphs: 1.
Abstract section exceeds max word limit: 150.
Abstract section contains the following legal terms: means

Claims Summary

Total Claims: 19
Independent Claims: 2
Dependent Claims: 17
Multiple Claims: 0
Deleted/Cancelled Claims: 0
Withdrawn Claims: 0

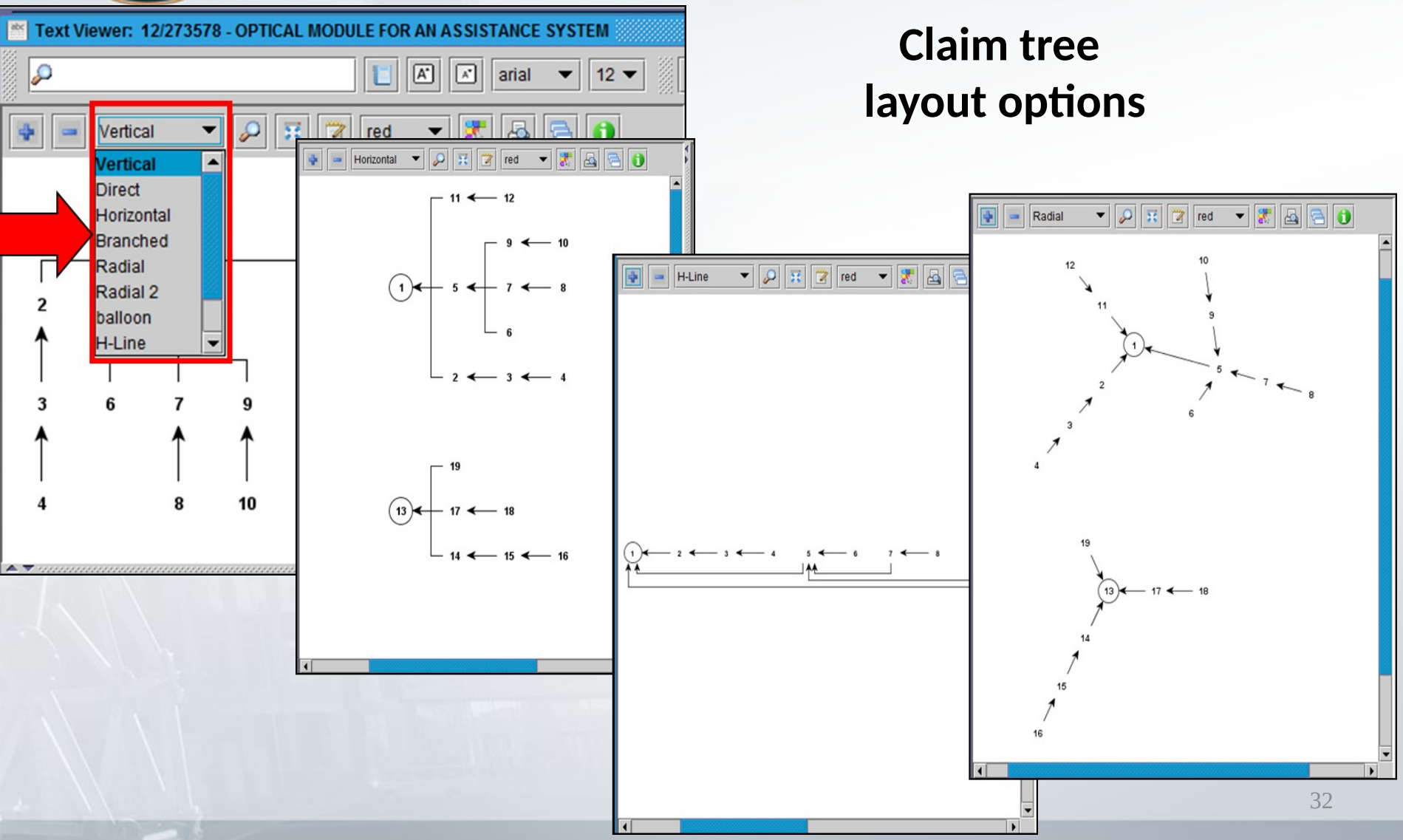
Results of Formalities Checks (con't)

CLAIMS TREE



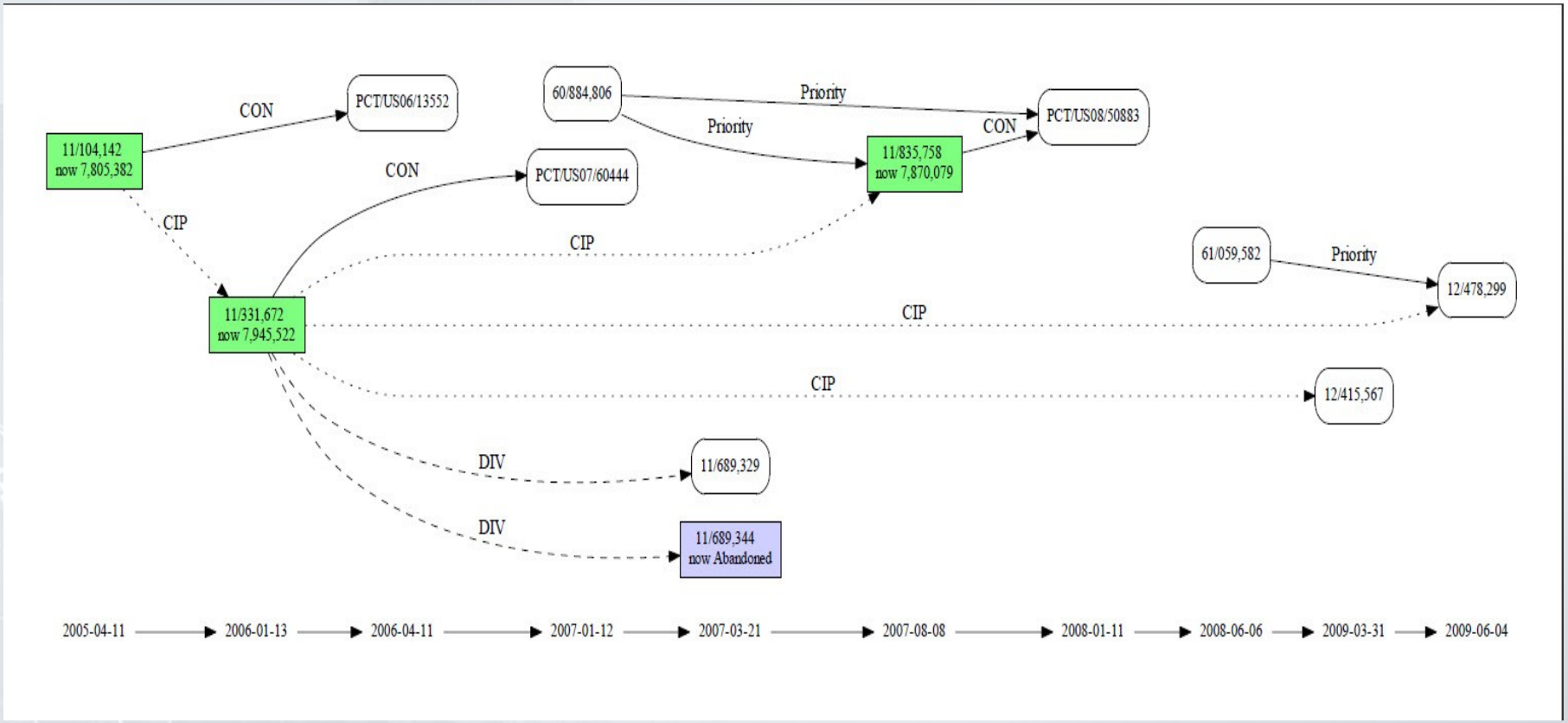
Results of Formalities Checks (con't)

Claim tree layout options



Results of Formalities Checks (con't)

Patent Family Map Showing Continuity



AMENDMENTS

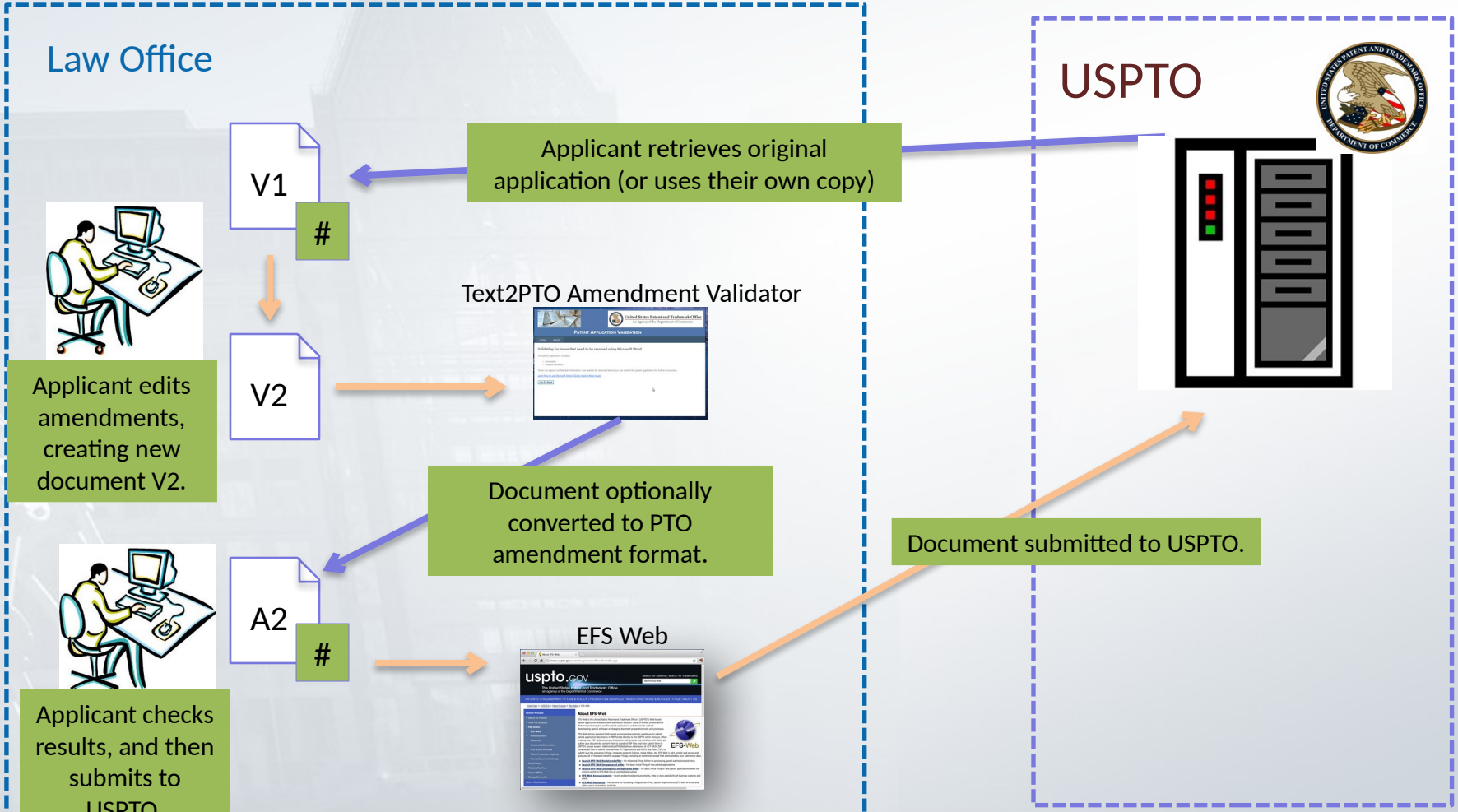
Modify a copy of your original document

Optional automated official markup.

Amendments

- Start with your submitted Word file
  - If you lost it, get a copy from the USPTO.
- Edit it with track changes on.
  - Delete and add material to the claims and specification.
- Run it through the Amendment Validator.
  - (Optional) It will automatically reformat the edits you did into the USPTO required format.
  - (Optional) It will generate a cover page and description of the changes.

Edit copy of original file to create amendments.



What Video Should Show

- Making the edits on claims for amendments [90 sec]
  - Show the user making a copy of the original document.
  - Show user deleting one claim, editing one claim, and adding one claim using track changes.
  - Show user uploading the document to the amendment validator.
  - Show the claims transformed into PTO format.
  - If possible, show the before and after side by side.

What Video Should Show

- Making the edits on the spec for amendments [60 sec]
  - Show the user adding and deleting text from the spec.
  - Show user uploading the document to the amendment validator.
  - Show the spec transformed into PTO format.
  - If possible, show the before and after side by side.

What Video Should Show

- Making the cover sheet for the amendments [90 sec]
  - Show the generated cover sheet for the changes made in the previous two segments.

ADDED VALUE: PARAGRAPH NUMBERING

Automatically added paragraph numbers; keeps track of version numbers

Paragraph Numbering

- The validator tool can automatically number the paragraphs.
- Additionally, these numbers can be augmented with the version number of the document.
  - Application version 1
  - Amendment version 3
- This allows for easier communication between the applicant and the examiner about changed material.

What Video Should Show

- [30 sec]
- Show text of original application with its paragraph numbers
- Then show editing of that document to create amendments.
- The numbering should automatically work if they start with the version that was uploaded to the PTO.

Benefits:
Filing Applications as Text

- Better for applicants:
  - No scanning
  - Re-use the text for amendments
  - Some formalities checks
- Better for examination:
  - Cost savings
  - Higher accuracy
  - Enables fantastic new tools for patent examiners.

WE WELCOME YOUR FEEDBACK!

What do you think?

TECHNICAL DETAILS

Some Technical Details

- The new patent examination tools make use of XML to represent documents in the system.
- Modern versions of word processing tools like Word store documents in OpenXML, an open format.
- USPTO can transform OpenXML documents into the format needed by the new tools.

More Technical Details

- Word allows certain kinds of markup that the USPTO does not want to see in the XML.
- The Validator Tool checks for this kind of markup and warns you.
  - You can remove the markup yourself,
  - Or let the Validator Tool remove it for you.

A Fragment of OpenXML

```
<w:body>
  <w:p w:rsidR="00407A0E"
       w:rsidRDefault="009519F9">
    <w:r>
      <w:t>Hello, World</w:t>
    </w:r>
    <w:bookmarkStart w:id="0"
                     w:name="_GoBack"/>
    <w:bookmarkEnd w:id="0"/>
  </w:p>
  <w:sectPr w:rsidR="00407A0E">
    <w:pgSz w:w="12240"
            w:h="15840"/>
```

Some Transformation Code

```
ass Program

static object TransformToSimpleXml(XNode node, string defaultParagraphStyleId)
{
    XElement element = node as XElement;
    if (element != null)
    {
        if (element.Name == W.document)
            return new XElement("document",
                element.Element(W.body).Elements()
                    .Select(e => TransformToSimpleXml(e, defaultParagraphStyleId)));
        if (element.Name == W.p)
        {
            string styleId = (string)element.Elements(W.pPr)
                .Elements(W.pStyle).Attributes(W.val).FirstOrDefault();
            if (styleId == null)
                styleId = defaultParagraphStyleId;
            return new XElement("p",
                new XAttribute("style", styleId),
                element.LogicalChildrenContent(W.r).Elements(W.t).Select(t => (string)t)
                    .StringConcatenate());
        }
        if (element.Name == W.sdt)
            return new XElement("contentControl",
                new XAttribute("tag", (string)element.Elements(W.sdtPr)
                    .Elements(W.tag).Attributes(W.val).FirstOrDefault()),
                element.Elements(W.sdtContent).Elements()
                    .Select(e => TransformToSimpleXml(e, defaultParagraphStyleId)));
        return null;
    }
    return node;
}
```

More Technical Details: Hash Codes as IDs

- It is standard practice to create a document "signature" from a hash function.
- [Details about which hash function we will be using should go here.]