Capítulo 2

Marco teórico

2.1. Notaria

2.1.1. Historia de la Notaria

La función notarial está plenamente relacionada con el uso, en la actualidad, de protocolos o libros, que es donde se asientan los actos jurídicos. Además, existen otras herramientas como son los sellos notariales, los demás sellos para las distintas operaciones jurídicas, la oficina, los apéndices, los índices, etc. [2]

Si nos remontamos a la Edad de Piedra sería imposible imaginar esta función ya que se tenía que satisfacer las necesidades mas elementales como la comida, el abrigo y la seguridad. Pero eso no implicaba que no se tomaran decisiones importantes, ya que estas decisiones eran tomadas en conjunto de manera oral, el lenguaje más antiguo, no en documentos como lo hacemos hoy en día. En el comienzo para que hubiera certeza en los actos que ocurrían se requería del conocimiento de toda la población quienes fungían como testigos y daban formalidad al acto jurídico. Posteriormente existió el registro en lápidas gigantes o en frescos que hacían del conocimiento de todos y brindaban certeza y seguridad de la historia.

El documento es un hecho producido por el natural vivir de la humanidad, el notario es una consecuencia necesaria para la redacción y valoración de los mismos. Esta figura se conoció antes como el escriba, una persona que fungía como redactor con conocimientos universales, cultura y apta para dicho trabajo. El escriba egipcio es entonces el antecedente más remoto del notario.

Las leyes de Hamburabi marcan claramente que si no existen testigos, es causa de nulidad en un contrato y por tanto la necesidad de un contrato, el cual, si no se celebraba, era causa de pena de muerte.

En Grecia, los síngrafos y los apógrafos fungían la función de notarios, y cuyos contratos debían ser inscritos en el Registro Público. En Roma, se utilizaban tablas enceradas para redactar documentos formales, pero para el envió de correspondencia se utilizaba la tabella, de donde se derivo el personaje que fungía como un asesor jurídico, pues redactaba documentos relacionados con la actividad privada, el tabellón. Ofreciendo, al igual que el notario hoy en día, asesoría jurídica, pero en este caso, no tenía ninguna relación con el estado. Este tenía sus funciones bien definidas en la sociedad, redactado contratos, testamentos y demandas. Actuando cercano a la proximidad del foro. Era una persona dedicada a la redacción de actos jurídicos y convenios entre particulares. Posteriormente al habito de archivarlos, esto se volvió en una actividad excesiva creando la necesidad de la creación de archivos públicos.

En el imperio Bisantino existió el Manual de Derecho, en el que se describe "la Corporación de Notarios o Tabularii". El acto del notario siempre a tenido la característica de actuar como Rogatario es decir, actuar al ruego de las partes. Con la existencia de esta corporación se conoce la existencia de un cuerpo colegiado.

Debido a la fuerte organización provincial de sus instituciones, la materia notarial sienta sus bases en España en el siglo XI, y subsiste en lo esencial hasta nuestros días. En el año de 1225 se comienza a dar número a los notarios, esto en Aragón, que tenía un Colegio de Notarios, llamado Colegio de Notarios del Número y Caja. Así mismo, los notarias de Valencias estaban dispensado de exigir firma de los contratantes, además, de requerir testigos. Antes de la conquista de España a México, se publicaron las Siete Partidas y las Leyes del Fuero Real. Esta ley habla del secreto profesional, el cual debe guardarse, por ejemplo, en los testamentos de la obligatoriedad de prestar el servicio, o sea no podía negarse el Notario a actuar. Estas leyes contienen lo siguiente "Ningún Notario sea osado de poner en las cartas otros testimonios, sino los que fueren adelante y presente quando amás las partes se avinieron pleyto ante él e le mandaren facer carta" también, contiene que "Ningún Escribano Público no faga carta entre ningunos homes, menos que los conoscer he de saber usu nombres, si fueren de la conscidos; e ningún escribano no meta otros escribanos en su lugar, mas cada uno faga las cartas con su mano; e si acaesciere que alguno de los Escribanos enfermare o por

otra razón no pudiera facer las cartas que ellos mandaran vayan algunos de los otros Escribanos Públicos que las fagan". Lo anterior nos muestra que a 700 años esto continua teniendo vigencia.

El Notario en el siglo XX, en la ley de 1901 se definen los siguiente principios: El principio de la forma, de la inmediación, de notoriedad, de matricidad o protocolo y de fe pública.

El principio de forma se refiere a las formas escritas documentales en las que interviene el notario, este principio se refiere a las formalidades que conlleva el acto notarial para llevar acabo dicho procedimiento, es decir, las formalidades indispensables para llevar acabo el acto. El principio de inmediación implica la presencia directa e inmediata de las personas y de las cosas. El principio de notoriedad implica la documentación de actos que el notario no presencia pero que puede conocer de manera indirecta. El principio de matricidad o protocolo implica que la ley regula todo. Finalmente, la fe pública se refiere a que los actos deben ser creídos para ser aceptado, por esto mismo se otorga a los Notarios un poder jurídico que tiene le efecto de fehaciencia.

En la ley del Notariado para el Distrito Federal y Territorios Federales del 22 de enero de 1932 se especifica el principio de metricidad, que habla sobre las características físicas de las hojas del protocolo.

2.1.2. Notaria en el estado de Puebla en la actualidad

Notario, como lo define la ley del Notariado del Estado de Puebla en su mas reciente publicación, es el profesional del Derecho investido de fe pública por el Estado, que por delegación ejerce una función de orden público, y que tiene a su cargo recibir, interpretar, redactar y dar forma legal a la voluntad de las personas que ante él acuden, y conferir autenticidad y certeza jurídica a los actos y hechos pasados ante su fe, mediante la consignación de los mismos en instrumentos públicos de su autoría. [3]

La fe pública de un notario tiene y ampara un doble contenido: dar autenticidad, fuerza probatoria y, en su caso, solemnidad a las declaraciones de voluntad de las partes que intervienen en las escrituras; y en en el caso de las actas y certificaciones, acredita la exactitud de lo que el Notario hace constar como lo percibió por sus sentidos.

La organización de las Notaría en México es atribuida al Estado a través del titular del Poder Ejecutivo de la Entidad, quien se encargará de la dirección y vigilancia del funcionamiento de estas por conducto de la Secretaría

de Gobernación.

El cargo de Notario es vitalicio. Sólo podrán ser suspendidos o cesados, en los términos previstos en la Ley del Notariado del Estado de Puebla.

El Despacho del Notario se denomina Notaría Pública. En Puebla, existe una Notaría en los Distritos Judiciales por cada veinticinco mil habitantes de la población económicamente activa, de conformidad con los censos poblacionales del Instituto Nacional de Estadística, Geografía e Informática (INEGI), además el Ejecutivo del Estado tomará en consideración las necesidades sociales, actividad económica y la actividad del propio servicio notarial.

El Notario, tiene a su cargo recibir, interpretar, redactar y dar forma legal a la voluntad de las personas que ante él acuden, conferir autenticidad y certeza jurídica a los actos y hechos pasados ante su fe. Además, un notario en México fungirá no sólo como auxiliar en la administración de la justicia, si no también como consejero o asesor de los comparecientes en materia jurídica, con el fin de explicarles el valor y las consecuencias legales, además de funcionar como árbitro o mediador, y expedirá a los interesados los testimonios, copias o certificaciones.

Los notarios no pueden autenticar actos o hechos cuyo contenido sea física o legalmente imposible, o su fin sea contrario a la Ley a las buenas costumbres.

En el estado de Puebla los Notarios tendrán carácter de Titulares, Auxiliares, Asociados o Suplentes. El Notario Titular o de Número es a aquel a cuyo favor se extiende por el Titular del Poder Ejecutivo del Estado la patente respectiva. El Notario Auxiliar es el designado por el Titular del Poder Ejecutivo del Estado a propuesta del Notario Titular, este tendrá las mismas funciones que el Titular, pero actuará en el mismo protocolo y sello del Titular y deberá hacer constar en los instrumentos su carácter de Auxiliar. Asociados se les llama a dos Notarios Titulares que se reúnen para actuar, ambos, en el del protocolo más antiguo. Notario Suplente es el que entra a sustituir las funciones del Titular que no tiene Auxiliar ni Asociado alguno en caso de falta temporal o definitiva, y esto, mientras no se asigne un nuevo Titular.

El Notario será el encargado en su actuación de la creación de las Escrituras Públicas, que son el instrumento en donde el notario asienta en el protocolo un acto o hecho jurídico autenticado con su sello firma. Además deberá ser firmado por los intervinientes en cada página. Además expresará el lugar, fecha, nombre y número de la Notaría, su calidad, hora del otorgamiento; nombre, fecha de nacimiento, estado civil, nacionalidad, profesión u

ocupación y domicilio de los intervinientes. También, será el encargado de la creación del Acta Notarial, instrumento público en el cual consigna hechos apreciados por medio de los sentidos del notario, que por su naturaleza no pueden calificarse como actos y contratos. En este caso bastará mencionar el nombre y apellido de la persona con quien se entienda el procedimiento o diligencia, sin necesidades de agregar sus demás generales.

Estos actos, en la actualidad, son registrados, en Puebla, en el Instituto Registral y Catastral del Estado de Puebla, el cual "basa su sistema registral en el folio electrónico, que es la unidad básica e intransferible en la que se concentra la información relativa a un determinado inmueble, mueble, persona jurídica o acto jurídico gubernativo, con el fin de crear un historial jurídico propio y único que arroje en un mismo momento de consulta la evolución del folio a partir de su creación", este sistema actualmente funciona como un registro manual pero "continuará vigente sólo en tanto se dicten las medidas para la transición al sistema de folio electrónico" cuyo reglamento "establecerá la forma y términos en que operarán esos sistemas. El Sistema Informático Registral se integra con la información que incorporan y validan los servidores públicos del Registro respecto a cada inscripción o anotación correspondiente a los folios electrónicos, con el propósito de tener capturada, almacenada y custodiada la información registral en una base de datos cuya consulta, reproducción y transmisión, sirve para la óptima operación del Registro." Es decir, este registro aun no ha quedado definido, el propósito del siguiente trabajo es definir un propuesta para llevar acabo el mismo.

2.2. Firma

2.2.1. Historia de la firma

La utilización de las firmas es muy antigua, de acuerdo a rabino Pinchas Allouche, en el episodio "Why Do We Sign For Things? A Rabbi, A Lawyer And A MasterCard Exec Explain" [4] del podcast Planet Money, de NPR, basado en la colección de libros del Talamud con más de mil años de antigüedad. En esta colección no sólo se mencionan las firmas, también se mencionan algunas reglas para estas. La primera regla es que "un garabato está prohibido", se debe poder leer el nombre de la persona, la segunda es que "la regla que se aplica a un centavo también se aplica a mil monedas de oro", es decir, todas las transacciones son consideradas grandes.

La utilización del nombre como firma era esencial para poder identificar quien había firmado, y por ejemplo, en la utilización de testigos, saber quienes habían sidos estos. Esto era útil para poder verificar los hechos simplemente buscando a las personas. Pero, en esta época el número de posibles personas era muy bajo, por lo que la utilización de este método si era posible.

Durante la edad media la utilización de sellos de cera era el método preferido para firmar documentos, posteriormente durante el renacimiento se comenzó a utilizar la firma como la conocemos. Hoy en día, muchas tiendas han dejado de pedir firmas, ya que estás no son útiles para uso comercial pues es imposible, como se hacía antes con los cheques personales, donde se tenían cuartos completos de personas revisando las firmas del cheque con las que se tenían en registro, revisar cada uno con las firmas archivadas en el registro del banco, de hecho, en la actualidad, nadie revisa las firmas de las diferentes transacciones comerciales que realizamos. Es por eso que muchas tiendas, hoy en día, no requieren firmas y algunas tarjetas de crédito comienzan a utilizar un PIN (Personal Identification Number, es decir, un número de identificación personal).

Las firmas, por ejemplo, pueden variar mucho de la que se firma en una identificación oficial como lo es la credencial para votar emitida por el Instituto Nacional Electoral (INE) que hoy en día se ocupa como método de identificación principal o el reverso de la tarjeta de crédito a lo que se firma por ejemplo en la realización de transacciones comerciales o en dispositivos de captura de firmas electrónicos. Estos últimos, por ejemplo, dice Heidi Harralson, una documentista forensica en una entrevista con The Atlantic para el artículo “Signing Off: The Slow Death of the Signature in a PIN-Code World” [5] no permiten capturar firmas útiles para análisis forénsicos.

En las Notarias, se requiere aun la firma de las personas como medio para identificarlas, además de los documentos que el notario solicite para poder realizar la operación. El problema es que una vez emitido el documento no hay una forma sencilla de poder validar que, en primera, las firmas grabadas en dicho documento sean verídicas y, en segunda, el sello notarial y firma del notario también lo sean. Es, más bien, la utilización de la firma digital, una posible solución pues esta puede ser autenticada con su contraparte pública (siendo la llave privada la firma personal), sin necesidad de que los actores estén presentes y más bien sólo verificando una operación real.

2.2.2. Historia de la firma digital

Con "la era del correo electrónico" [6] R.L. Rivest, A. Shamir, y L. Adleman indican que "debemos asegurar que dos propiedades importantes del sistema actual de 'correspondencia en papel' se conserven: (a) que los mensajes sean privados y (b) que los mensajes se puedan firmar." en el método propuesto por estos se describe un sistema capaz de realizar esto basado en un "ecosistema de llave pública", concepto inventado por Diffie y Hellman. "En un ecosistema de llaves públicas cada usuario coloca en un archivo público un proceso de encriptación E [y] el usuario guarda en secreto los detalles del procedimiento de desenciprción D".

Hablamos entonces del método RSA por los apellidos de sus autores (Rivest, Shamir y Adleman) del cuál se hablará más a detalle posteriormente. El algoritmo RSA fue descrito por primera vez en 1977. Pero la idea como lo mencionamos anteriormente es atribuída a Diffie y Hellman que publicaron el concepto en 1976 donde, en su formulación, se crea un secreto compartido, es decir, se acuerda un número y una base, cada uno de los actores, es decir, Alicia (A) y Beto (B) escogen un número entero y mediante el algoritmo propuesto por Deffie y Hellman se genera un nuevo número que llamaremos secreto compartido. Este sistema dejó abierta la posibilidad de la creación de una opción de ecosistema de llaves públicas de una sola vía, es decir una opción donde solamente uno de los dos usuarios pudiera enviar el mensaje y solamente el otro pudiera ver el mismo.

El algoritmo RSA, entonces, funciona mediante la publicación de la llave pública generada a partir de la llave privada del usuario A (Alicia), utilizando esta llave pública B (Beto) encrípta el mensaje que únicamente Alicia utilizando su llave privada puede desencríptar.

Es entonces la combinación de esta llave pública y llave privada la firma digital de Alicia ya que solamente ella puede abrir el mensaje enviado por Beto. Finalmente, el algoritmos RSA también puede funcionar utilizando la llave privada para encriptar el mensaje y la llave pública para desencriptarlo, esto con la finalidad de certificar que solamente Alicia creó ese mensaje y que cualquiera pueda comprobar eso tal como lo hace el algoritmo SSL.

2.3. Facturación electrónica

La facturación electrónica sentó precedente en México para la aceptación y emisión de documentos digitales. Hoy en día, el modelo utilizado para la facturación digital en México es utilizado en otros actos como lo son la emisión de documentos oficiales a través de internet certificados por una autoridad. Actualmente, el modelo utilizado por la facturación electrónica en México ha penetrado otras áreas distintas a la fiscal y ha permitido ser un punto de acceso a la aceptación de los documentos electrónicos y digitales como comprobantes de actos o hechos en México.

2.3.1. Historia de la facturación electrónica en México

La factura electrónica o Comprobante Fiscal Digital (CFD) es como lo menciona el contador público Lauro E. Arias de la Comisión Representativa del Instituto Mexicano de Contadores Públicos ante las Administraciones Generales de Fiscalización del SAT en el artículo titulado "Factura Electrónica Una Realidad en Seis Meses" [7] es un archivo "que tiene la misma validez que la de un impresor autorizado" donde "ambas sirven para comprobar la realización de transacciones comerciales entre comprador y vendedor, ya sea por un bien o servicios y [que] obliga a realizar el pago correspondiente, de acuerdo con lo establecido con la factura". También, menciona que esta debe cumplir con ciertos requisitos que revisaremos con detalle más adelante.

Según el contador público Arias, la historia de la facturación electrónica en México deriva "del esfuerzo de personas y organismos que deseaban su aplicación en nuestro país". Iniciando su historia en 1997 donde la iniciativa privada "previendo la necesidad de una factura electrónica" buscó un esquema legal que permitiera su uso instuyendo "un Comité de Factura Electrónica" que se conformó por "alrededor de 45 empresas asociadas a la Asosiación Mexicana de Comercio Electrónico".

Se desarrolló entonces "un modelo y varias pruebas pilotos" que permitieron "identificar los requerimientos de modificaciones a las leyes para establecer el marco jurídico [que] llevara a la implementación de la factura digital". Siendo así en Mayo de 2004 cuando el SAT aprobó la factura electrónica como un medio de comprobación fiscal en el Anexo 20 de la Miscelánea Fiscal, donde se establecen todos los fundamentos legales y técnicos para cualquier persona física o moral que desee emitir Comprobantes Fiscales Digitales.

2.3.2. Firma digital

Una firma digital o firma electrónica es, según el SAT [8] "un archivo digital que te identifica al realizar trámites por internet en el SAT e incluso en otras dependencias de Gobierno de la República". La firma electrónica proporcionada por el SAT "es única, es un archivo seguro y cifrado que incluye tu firma caligráfica" y "por sus características es segura y garantiza tu identidad".

Una firma digital es un esquema matemático para demostrar la autenticidad de un mensaje digital o documento. Una firma digital valida da al receptor de la misma la certeza de que el que la envía no puede negar la generación del mismo y que el mensaje o documento no fue alterado.

2.3.3. Esquema de facturación electrónica en México

El esquema de facturación se encuentra plasmado en el Anexo 20 de la Miscelánea Fiscal, aunque dicho anexo fue aprobado en el año 2004 por el SAT a continuación veremos las características de este esquema de acuerdo a la Modificación al Anexo 20 de la Resolución Miscelánea Fiscal para 2006. [9]

De acuerdo al Anexo 20 de la Miscelánea Fiscal mencionado anteriormente donde se definen las "Características técnicas del archivo que contenga el informa mensual de comprobantes fiscales digitales emitidos", los "estándares y especificaciones técnicas que deberán cumplir las aplicaciones informáticas para la generación de claves critográficas asimétrica[s] a utilizar para [la] Firma Electrónica Avanzada", el "estándar de comprobante[s] fiscal[es] extensible", la "generación de sellos digitales para comprobantes fiscales digitales", el "uso de la facilidad de ensobretado" y el "uso de la facilidad de nodos". En la sección B donde se definen los "estándares y especificaciones técnicas que deberán cumplir las aplicaciones informáticas para la generación de claves de criptografía asimétrica a utilizar para Firma Electrónica Avanzada" las aplicaciones informáticas de las que "el contribuyente se auxilie para la generación de su par de claves (clave pública y privada)" se especifica que las llaves públicas y privadas son "de tipo RSA de 1024 bits" y esta se almacenará en "un archivo configurado de acuerdo al estándar PKCS8 en formato DER". DER, por sus siglas inglés (Distinguished Encoding Rules) son una serie de reglas de codificación que permite la codificación de valores, que serán firmados, de una sola manera.

Además se define que los certificados o denominados en el Anexo 20 de la Miscelánea Fiscal "requerimientos digitales" deberán contener la llave pública y se encontrarán en "el estándar PKCS10 en formato DER", deberá este certificado "para el procesamiento adecuado del requerimiento digital" o certificado contener el Registro Federal de Contribuyente (RFC) a 12 posiciones en el caso de personas morales y a 13 posiciones para personas físicas exceptuando casos especiales especificados en el Anexo 20 de la Miscelánea Fiscal, correo electrónico, clave de revocación (obtenido mediante la unión del RFC en mayúsculas y la clave de revocación proporcionada por el contribuyente, valor al cual se le aplica el algoritmo SHA-1 y es expresada en Base 64), adicionalmente se deberá incluir el CURP de la persona física o el CURP del representante legal en el caso de personas morales.

En el caso del comprobante fiscal digital a través de Internet o CFDI nos referiremos al Anexo 20 de la Segunda Resolución de Modificaciones a la Resolución a la Miscelánea Fiscal para 2015 [10], publicada el 14 de mayo de 2015 donde se especifica el "estándar de comprobante fiscal digital a través de internet", se habla "del comprobante fiscal digital a través de Internet que ampara retenciones e información de pagos" y finalmente "de los distintos medios de comprobación digital". Sobre el estándar del CFDI se especifica como formato electrónico único la utilización del XML Schema Definition (XSD) [11, 12] presentado en la primera sección del Anexo 20 de la Segunda Resolución de Modificaciones a la Resolución a la Miscelánea Fiscal para 2015, donde XSD es una recomendación que especifica como describir formalmente los elementos en un documento XML (Extensible Markup Language), "validando su forma y sintaxis en un archivo con extensión XML, siendo este el único formato para poder representar y almacenar comprobantes de manera electrónica o digital". Podemos hallar entonces en el XSD definido un estándar que contiene los siguientes campos enumerados de acuerdo a como aparecen en el estándar: Emisor, que define "la información del contribuyente emisor del comprobante", dentro de este campo se encuentra la información sobre la persona física o moral que emite el CFDI, los datos dentro de este campo incluyen de manera obligatoria el régimen fiscal que incorpora "los regímenes en los que tributa el contribuyente emisor" y que bajo el atributo régimen puede contener más de un régimen y el atributo RFC donde se registra la Clave del Registro Federal de Contribuyentes "correspondiente al contribuyente emisor del compobante sin guiones o espacios"; receptor, que define "la información del contribuyente receptor del comprobante" y que deberá contener el RFC al igual que en el emisor; conceptos, que enlista "los

conceptos cubiertos por el comprobante"; impuestos que "captura los impuestos aplicables", la versión que contiene el "valor prefijado a 3.2 que indica la versión del estándar bajo el que se encuentra expresado el comprobante"; la fecha que expresa "la fecha y hora de expedición del comprobante fiscal" y que se expresa de acuerdo a la especificación "ISO 8601"; sello que contiene "el sello digital del comprobante fiscal, al que hacen referencia las reglas de resolución miscelánea aplicable" y que deberá "ser expresado como una cadena de texto en formato Base 64"; la forma de pago "para precisar la forma de pago que aplica"; el número de certificado "para expresar el número de serie del certificado de sello digital que ampara el comprobante, de acuerdo al acuse correspondiente a 20 posiciones otorgado por el sistema del SAT"; subtotal donde se representa "la suma de los importes antes de descuentos e impuestos"; total que representa "la suma del subtotal, menos los descuentos aplicables, más los impuestos trasladados, menos los impuestos retenidos"; el tipo de comprobante que expresa "el efecto del comprobante fiscal para el contribuyente emisor"; el método de pago "para expresar el método de pago de los bienes o servicios amparados por el comprobante"; y finalmente el lugar de expedición donde se incorpora "el lugar de expedición del comprobante".

Finalmente, sobre la generación del sello digital se especifica que se deberá "aplicar el método de digestión SHA-1 a la cadena original a sellar incluyendo los nodos Complementarios" para posteriormente "con la clave privada correspondiente al certificado digital del emisor del mensaje y del sello digital, encriptar la digestión del mensaje obtenida en el paso [anterior] utilizando para ello el algoritmo de encripción RSA" y finalmente codificarlo en Base 64 de la cual se hablará más a detalle posteriormente.

2.4. Criptografía

2.4.1. RSA

Especificado en el Anexo 20 de la Segunda Resolución de Modificaciones a la Resolución Miscelánea Fiscal para 2015, RSA es el método utilizado para la generación de sellos digitales para Comprobantes Fiscales Digitales a través de Internet (CFDI). En esta resolución se especifica que "RSAPrivateEncrypt que utiliza la clave privada del emisor para encriptar la digestión del mensaje" y "RSAPublicDecrypt, que utiliza la clave pública del emisor para desencriptar el mensaje" es el método utilizado para la creación de

los sellos digitales. Así mismo, como ya se mencionó previamente las claves utilizadas deberán ser tipo RSA de 1024 bits.

RSA, que por las iniciales del apellido de sus creadores (Rivest, Shamir y Adleman) es un método de encriptación "con la noble propiedad que revelar públicamente una llave de encriptación no revela la llave de desencripción correspondiente", es decir, no se necesitan modos seguros de transmitir las llaves ya que el mensaje se encripta utilizando una llave públicamente revelada por el receptor del mensaje, que será el único que podrá desencriptar el mensaje ya que es el único que conoce la llave para realizar esta operación. Por otra parte, este método también permite que "un mensaje sea firmado usando una llave privada de desencripción" ya que "cualquiera puede verificar la firma usando la llave públicamente revelada de desencripción", esto debido a que "las firmas no pueden ser falsificada, y el que firma no puede negar posteriormente la validez de la firma".

Antes de continuar describiendo el funcionamiento interno del algoritmo propuesto por Rivest, Shamir y Adleman debemos comentar la importancia y la utilización del mismo para la firma de documentos. Para poder entonces, ser aceptada la firma de un documento el receptor del mismo debe poder probar que el mensaje fue creado por el remitente. Es entonces que "una firma electrónica debe ser dependiente del mensaje, además de dependiente del firmante" de otra forma el mensaje "podría ser modificado por el remitente", ya que como lo mencionan Rivest, Shamir y Adleman también podría ser agregada a cualquier otro mensaje ya que "es imposible detectar el copiado y pegado electrónico".

Es entonces, el funcionamiento del firmado electrónico como se explica en el Anexo 20 de la Segunda Resolución de Modificaciones a la Resolución Miscelánea Fiscal para 2015 el siguiente: Beto envía a Alicia un mensaje firmado en un ecosistema de llaves públicas, para esto deberá procesar el mensaje utilizando su llave privada con el que obtendrá la firma del mismo para posteriormente enviar la firma, no es necesario enviar el mensaje ya que este puede ser obtenido a través de la firma, Alicia entonces utilizando la llave publicada por Beto puede obtener el mensaje con lo que ahora ella posee la firma del mismo y el mensaje, Beto no puede negar haber creado el mensaje ya que el es el único que pudo haber creado un mensaje que puede ser abierto utilizando su llave pública. Para esto se presupone que la llave pública siempre es accesible y que, aunque es posible falsificar esta información en una red pública como internet, la llave pública no se ha falsificado.

$$C \equiv E(M) \equiv M^e \pmod{n} \tag{2.1}$$

$$D(C) \equiv C^d \pmod{n} \tag{2.2}$$

El funcionamiento interno del método de encripción propuesto por Rivest, Shamir y Adleman (RSA) es entonces, para la encripción de un mensaje transformado a bloques de número enteros entre 0 y $n-1$, el cifrado del mensaje elevándolo a la e-ésima potencia modulo n. Para la descencripción se deberá elevar a la potencia d, modulo n. Son entonces los algoritmos para encripción y desencripción los siguientes: (2.1) para el mensaje, (2.2) para el texto cifrado. Dónde en (2.1) el mensaje M a la e-ésima potencia módulo n es equivalente al mensaje M encriptado E y equivalente a el mensaje M cifrado C. Y dónde en (2.2) el mensaje M a la d-ésima potencia modulo n es equivalente al mensaje M desencriptado D.

Es importante notar que "la encripción no aumenta el tamaño del mensaje" debido a que "tanto el mensaje como el texto cifrado son enteros entre 0 y $n-1$.

$$n = p \cdot q \tag{2.3}$$

Para la creación de la llaves, que son pares de enteros positivos (e, n) para la llave de encripción o llave pública y (d, n) para la llave de desencripción, se deberá crear primero n que es el producto de dos primos p y q como vemos en (2.3).

$$gcd(d, (p-1) \cdot (q-1)) = 1 \tag{2.4}$$

$$e \cdot d \equiv 1 \pmod{(p-1) \cdot (q-1)} \tag{2.5}$$

Siendo los número primos muy grandes y aleatorios, y aunque n será público debido al tamaño de p y q estos estarán ocultos, "debido a la dificultad de factorizar n". Para la elección de d y e se deberán satisfacer (2.4) y (2.5). gcd en (2.4) es el máximo común divisor, que es el mayor divisor entero que divide tanto a d como a $(p-1) \cdot (q-1)$ y da como resultado de la división, en este caso, 1. e en (2.5) es creado a partir de p, q y d siendo este el inverso multiplicativo de d, modulo $(p-1) \cdot (q-1)$. $\equiv$ es la equivalencia, es decir, tienen el mismo valor, son equivalentes.

2.4.2. SHA-256

SHA es como lo indica el Anexo 20 de la Segunda Resolución de Modificaciones a la Resolución Miscelánea Fiscal para 2015 "una función hash (digestión, picadillo o resumen) de un solo sentido tal que para cualquier entrada produce una salida compleja de 160 bits (20 bytes)", cuya función es en el Anexo 20 de la Segunda Resolución de Modificaciones a la Resolución Miscelánea Fiscal para 2015 el "aplicar el método de digestión SHA-1 a la cadena original a sellar" con la finalidad de generar "una salida de 160 bits (20 bytes) para todo el mensaje". Finalmente, indica que "la posibilidad de encontrar dos mensajes distintos que produzcan una misma salida es de 1 en 2160, y por lo tanto en esta posibilidad se basa la inalterabilidad del sello, así como su no reutilización". Esta digestión se realiza previo a la firma del mensaje utilizando RSA, es decir, se realiza la firma sobre la salida del método de digestión.

El algoritmo SHA [13], Secure Hash Algorithm por sus siglas en inglés, fue como lo indica Penard [14] "desarrollado para la NIST [National Institue of Standards and Technology en asociación con la NSA [National Security Agency] y publicados por primera vez en Mayo de 1993 como el Estándard de Hash Seguro", postriormente "la primera revisión de este algoritmo fue publicada en 1995 debido al hallazgo de un error no publicado, y fue nombrado SHA-1". Además del hash SHA-1, la NIST otras funciones de hash más complejas llamadas SHA-224, SHA-256, SHA-384 y SHA-512 "algunas veces refericos como SHA-2". Debido a la similaridad de las funciones de hash SHA-2 con los algorítmos SHA-1, los cuales en 2005 probaron tener ataques por colisión de complejidad 2^{63} menor a la complejidad 2^{80} de SHA-1, en noviembre de 2007 la NIST se publicó la competencia para la creación de las funciones de hash SHA-3.

Se muestra el algorítmos SHA-256 en la figura 2.1. Donde M es el mensaje que primero se deberá asegurar que sea de largo $L < 2^{64}$, en caso de no ser así se agregará 1 al final del mensaje, por ejemplo, sea "01010000" el mensaje, se agregará como "010100001". Posteriormente se agregarán K "0"s donde K es el valor más pequeño no negativo que resuelve $L+1+K \equiv 448 \pmod{512}$ finalmente se agrega el bloque de 64-bits que representa L en representación binaria. Además, se utilizarán las funciones de la figura 2.2 donde cada una opera en palabras de 32-bits, representadas como x, y y z, donde $ROTR^n(x)$ es la rotación a la derecha, $ROTL^n(x)$ es rotación a la izquierda y $SHR^n(x)$ es desplazamiento a la derecha y $SRL^n(x)$ es desplazamiento a la izquierda,

```
For i = 1 to N

   1. Prepare the message schedule W:
      For t = 0 to 15
         Wt = M(i)t
      For t = 16 to 63
         Wt = SSIG1(W(t-2)) + W(t-7) + SSIG0(t-15) + W(t-16)

   2. Initialize the working variables:
      a = H(i-1)0
      b = H(i-1)1
      c = H(i-1)2
      d = H(i-1)3
      e = H(i-1)4
      f = H(i-1)5
      g = H(i-1)6
      h = H(i-1)7

   3. Perform the main hash computation:
      For t = 0 to 63
         T1 = h + BSIG1(e) + CH(e,f,g) + Kt + Wt
         T2 = BSIG0(a) + MAJ(a,b,c)
         h = g
         g = f
         f = e
         e = d + T1
         d = c
         c = b
         b = a
         a = T1 + T2

   4. Compute the intermediate hash value H(i):
      H(i)0 = a + H(i-1)0
      H(i)1 = b + H(i-1)1
      H(i)2 = c + H(i-1)2
      H(i)3 = d + H(i-1)3
      H(i)4 = e + H(i-1)4
      H(i)5 = f + H(i-1)5
      H(i)6 = g + H(i-1)6
      H(i)7 = h + H(i-1)7
```

Figura 2.1: Algoritmo SHA-256

```
CH( x, y, z) = (x AND y) XOR ( (NOT x) AND z)

MAJ( x, y, z) = (x AND y) XOR (x AND z) XOR (y AND z)

BSIG0(x) = ROTR^2(x) XOR ROTR^13(x) XOR ROTR^22(x)

BSIG1(x) = ROTR^6(x) XOR ROTR^11(x) XOR ROTR^25(x)

SSIG0(x) = ROTR^7(x) XOR ROTR^18(x) XOR SHR^3(x)

SSIG1(x) = ROTR^17(x) XOR ROTR^19(x) XOR SHR^10(x)
```

Figura 2.2: Operaciones para SHA-256

```
428a2f98 71374491 b5c0fbcf e9b5dba5
3956c25b 59f111f1 923f82a4 ab1c5ed5
d807aa98 12835b01 243185be 550c7dc3
72be5d74 80deb1fe 9bdc06a7 c19bf174
e49b69c1 efbe4786 0fc19dc6 240ca1cc
2de92c6f 4a7484aa 5cb0a9dc 76f988da
983e5152 a831c66d b00327c8 bf597fc7
c6e00bf3 d5a79147 06ca6351 14292967
27b70a85 2e1b2138 4d2c6dfc 53380d13
650a7354 766a0abb 81c2c92e 92722c85
a2bfe8a1 a81a664b c24b8b70 c76c51a3
d192e819 d6990624 f40e3585 106aa070
19a4c116 1e376c08 2748774c 34b0bcb5
```

Figura 2.3: Constante K para SHA-256

x es una palabra de w-bits y n es un entero con $0 <= n < w$. Finalmente, se definen las constantes $K0, K1, \cdots, K63$ en hexadecimal de izquierda a derecha en la figura 2.3, que representan los primeros treinta y dos bits de las partes fraccionarias de las raíces cúbicas de los primeros sesenta y cuatro números primos [15].

Se inicia entonces el algorítmo en la figura 2.1 con la preparación de W, posteriormente se inician las variables, se realizan las operaciones para el computo del hash y se calcula el valor intermedio del hash $H(i)$ para cada i de 0 a N, donde N es el número de bloques. Al finalizar el algorítmos en la figura 2.1 se concatena el resultado de $H(N)0, H(N)1, \cdots, H(N)7$.

2.4.3. Base 64

De acuerdo al RFC (Request For Comments) 4468 [16], la codificación base64, en donde también se discuten las especificaciones de la base16 y base32, se comenta que la codificación utilizando bases se utiliza "en muchas situaciones para guardar o transferir datos en ambientes que, probablemente por razones de compatibilidad, están restringidos a datos en formato US-ASCII", además especifica que no necesariamente se deba razones de compatibilidad la transformaciones si no "para hacer posible la manipulaciones de los objetos en editores de texto". Uno de los defectos de la base64 y de los lenguajes utilizando bases es que debido a que utilizan un alfabeto reducido, caracteres no pertenecientes al mismo pueden existir, como son los emojis hoy en día, y ser utilizados para transferir información corrompida o corrupta.

De acuerdo a la especificaciones, debido a que en algunas circunstancias no se puede saber el tamaño de los datos a transferir, es entonces que se utiliza el relleno especificado por el signo de igual ("="). En cualquier caso, se debe incluir al final de la cadena codificada el relleno. Además, el alfabeto elegido en el caso de la base64 deberá vigilar la utilización de la diagonal ("/") y el símbolo de suma ("+") debido a que por razones de compatibilidad estos se pueden utilizan para otros fines.

La Base 64 está diseñada para "representar secuencias arbitrarias de octetos en una forma que permita el uso de letras en mayúscula y minúscula pero que sean leíbles por el humano", es entonces que se utiliza un subconjunto de 65 caracteres del US-ASCII que permite representar 6 bits por cada uno. Podemos ver en la figura 2.4 el alfabeto en base64 especificado en el RFC 4468, donde cada valor de la columna izquierda es equivalente al valor de 6 bits de nuestra entrada que dará como salida el valor codificado en la columna

Value	Encoding	Value	Encoding	Value	Encoding	Value	Encoding
0	A	17	R	34	i	51	z
1	B	18	S	35	j	52	0
2	C	19	T	36	k	53	1
3	D	20	U	37	l	54	2
4	E	21	V	38	m	55	3
5	F	22	W	39	n	56	4
6	G	23	X	40	o	57	5
7	H	24	Y	41	p	58	6
8	I	25	Z	42	q	59	7
9	J	26	a	43	r	60	8
10	K	27	b	44	s	61	9
11	L	28	c	45	t	62	+
12	M	29	d	46	u	63	/
13	N	30	e	47	v		
14	O	31	f	48	w	(pad)	=
15	P	32	g	49	x		
16	Q	33	h	50	y		

Figura 2.4: Alfabeto Base 64

derecha, por ejemplo, si tenemos la cadena 000000 como entrada, esta será representada por el valor A en la salida. El último valor ("=") representa el relleno en caso de no contar con una cadena de 24-bits como entrada como se explica a continuación.

El proceso de codificación representa "grupos de 24-bits como entrada y 4 caracteres codificados como salida". Procediendo de izquierda a derecha se crean grupos de 24-bits al juntar 3 grupos de 8-bits que serán utilizados como entrada. Estos 24 bits son tratados entonces como 4 grupos concatenados de 6-bits, esto debido a que 4 grupos de 6-bits representan 24-bits al igual que 3 grupos de 8-bits permitiendo así la transformación, cada uno de estos grupos representado entonces por un carácter del alfabeto de la base64, en caso de existir menos de 24-bits dentro del subconjunto final, se realizan tres tipos de procedimientos: en caso de que la entrada sea exactamente 8 bits, la última unidad de salida codificada será "dos caracteres seguidos de dos caracteres de relleno; en caso de que la entrada sea exactamente 16 bits, la última unidad de salida "será de tres caracteres seguidos de un carácter de relleno"; en caso de que sea un múltiplo entero de 24 (2, 4, 6, 8, 12), la unidad de salida será "un integral múltiplo de 4 caracteres sin caracteres de relleno".

2.5. ECMAScript

2.5.1. JavaScript

ECMAScript es un lenguaje de scripting registrado y estandarízado por Ecma International[17].

JavaScript es un un lenguaje de programación interpretado con capacidades orientadas a objetos. "Sintácticamente, el lenguaje JavaScript base es parecido a C, C++, y Java" [18]. Sin embargo, en JavaScript no es necesario definir el tipo de las variables y constantes.

JavaScript era anteriormente principalmente usado en "navegadores web, y, en ese contexto, el propósito general es interactuar con el usuario, controlar el navegador web, y alterar el contenido de los documentos que aparecen en la ventana del navegador web" [19]. A este tipo de implementación JavaScript se le conoce como client-side, es decir, que es ejecutada en el lado del cliente y no en un servidor web. Existe, otra implementación para servidores web conocida como Node.js de la que se hablará más a fondo en la siguiente sección.

Los objetos en JavaScript son similares a las a tablas de hash, ya que ligan los nombres de las propiedades a sus valores o funciones. Además, la herencia es manejada a través de prototipos.

JavaScript soporta números, cadenas y booleanos como tipos de datos primitivos, además permite el uso de arreglos, fechas y expresiones regulares.

Aunque JavaScript es un lenguaje interpretado, y por lo mismo considerado un lenguaje de scripting, este es un lenguaje de programación completo igual de complejo que cualquier otro lenguaje.

2.5.2. Node.js

Node.js o Node es un ambiente JavaScript para servidores, basado en el tiempo de ejecución V8 de Google, está implementado en C y C++ "enfocandose en el rendimiento y bajo uso de mejoria" [20].

Node no depende en el uso de varios hilos para procesar peticiones concurrentes y más bien "se basa en un modelo I/O asíncrono", es decir, se basa en eventos que son delegados a otros hilos para ser procesados, siendo así un lenguaje no bloquedante para la realización de operaciones asíncronas que pueden tomar mucho tiempo como lo son los procesos de escritura y lectura de archivos, base de datos, entre otros. Por ejemplo, en caso de que un usua-

rio desee subir un archivo o una funcion requiera realizar una acción esta puede llamar a un evento específico.

2.5.3. JSON

JavaScript Object Notation o Notación de Objetos JavaScritp (JSON), es un formato independiente al lenguaje basado en texto para el intercambio de información derivado del estándar ECMAScript especificado en RFC 4627 [21].

JSON "puede representar cuatro tipos de objetos primitivos (cadenas, números, booleanos y nulos) y dos tipos estructurados (objetos y arreglos).

La finalidad de JSON era que fuera "minimalista, portable, textual, y un subconjunto de JavaScript".

JSON se ha utilizado "para el intercambio de datos entre aplicaciones escritas en todos estos lenguajes de programación: ActionScript, lenguajes C, ColdFusion, Common Lisp, E, Erlang, Java, JavaScript, Lua, Objective CAML, Perl, PHP, Python, Rebol, Ruby, y Scheme".

2.6. Herramientas y metodologías

2.6.1. Integración continua

La Integración Continua o Continuos Integartion (CI), [22] "es una práctica de desarrollo de software donde los miembros de un equipo integran su trabajo de forma continua" cada integración es entonces "verificada por un sistema automatizado de construcción (incluyendo las pruebas) para detectar errores en la integración lo más pronto posible".

El flujo de trabajo con la integración continua es tras haber obtenido una copia del código fuente de un software funcional y realizar los cambios deseados, se realizan pruebas locales sobre esta nueva pieza de software, una vez realizadas las pruebas exitosamente, debido a que en un equipo distintos desarrolladores pueden estar trabajando en distintas partes del software, se realiza nuevamente la obtención de una copia actualizada del código fuente principal y realiza la combinación automática de ambas copias, en caso de existir algún conflicto en la combinación se deberá solucionar para posteriormente realizar nuevamente las pruebas, esto se realiza hasta que se tenga una copia del código fuente principal más actualizado combinado con el código

creado y las pruebas locales realizadas sean exitosas. Posteriormente, se publica el código con los cambios realizados, debido a que la realización de una prueba local no indica que el software funcionará en otros ambientes se realiza una nueva prueba en un ambiente de integración continua, el resultado es entonces "una pieza de software que funciona correctamente", reduciendo el tiempo de producción ya que ahora todos los desarrolladores cuentan con una base estable y reduciendo el tiempo para encontrar bugs debido a que estos surgen rápidamente. Todo se realiza con la combinación de herramientas como Jasmine, para la realización de pruebas; Travis CI, como sistema de integración continua; y Git, como sistema manejador de código fuente.

Travis CI [23, 24], es un sistema de integración continua distribuido de código abierto que "permite a proyectos de código abierto registrar su repositoro en GitHub y realizar sus suites de pruebas" de manera distribuida, además, permite publicar dentro de nuestro proyecto en GitHub el estado de las pruebas realizadas y en caso de ser fallidas avisa al usuario responsable.

2.6.2. Desarrollo basado en pruebas

El desarrollo basado en pruebas o test driven development (TDD) es una práctica de desarrollo de software donde se desarrollan unidades de distintos caso a probar "antes de la implementación del código" [25]. Aunque el desarrollo basado en pruebas ha existido por varias décadas y ha sido utilizado esporádicamente, recientemente ha ganado visibilidad debido a su utilización y como "una práctica de Extreme Programming". Se desarrollan entonces unidades de prueba para un objeto o caso de uso, una importante regla del TDD es "si no puedes escribir una prueba para el código que vas a escribir, entonces no deberías pensar en escribir ese código". Posteriormente, estas pruebas llaman disparadores de los distintos objetos o casos de uso para ser probados esperando, entonces, un resultado que se compara con el resultado esperado definido en la unidad de prueba. [26]

Jasmine, es un framework de pruebas para JavaScript, este framework realiza las pruebas en el flujo de trabajo de integración continua. Basado en unidades de prueba como ScrewUnit, JSSpec, JSpec y RSpec, fue diseñado con principios sobre el buen desarrollo de pruebas en JavaScript con reglas como "no deberá estar ligado a ningún navegador, framework, plataforma o lenguaje" [27], "deberá trabajar en cualquier lugar donde JavaScript funcione", "no deberá ser intrusivo en la aplicación", entre otros. Desarollado en 2010 por Pivotal Labs, fue el sucesor de JsUnit [28].

2.6.3. Sistema de control de versiones

Los sistemas de control de versiones, sistemas de control de revisiones o sistemas de control de código son sistemas "esenciales para coordinar el trabajo en proyectos de software" [29]. El funcionamiento de un sistema de control de versiones busca solucionar el escalamiento de del desarrollo de software a un gran número de desarrolladores distribuidos en distintas ubicaciones.

Estas herramientas determinan en gran parte "como se coordina el desarrollo de nuevas características de un software, que tan continuamente distintas líneas de código son combinadas, como se realiza la revisión del código y como se organiza el soporte del código ya liberado".

"Las primeras herramientas de control de revisión automatizada estaban destinados a ayudar a un solo usuario para administrar revisiones de un solo archivo", hoy en día las herramientas han ampliado su alcance, ayudando a varias o miles de personas a trabajar en miles de archivos al mismo tiempo. [30]

Una de las herramientas más conocidas en la actualidad para control de versiones es Git, surgido en 2005 tras la controversia del equipo de desarrollo de Linux dirigido por Linus Torvals con BitKeeper, una de las perimeras herramientas de este tipo.

Hoy en día, GitHub [31], "con más de 12 millones de usuarios y 31 millones proyectos" es un servidor Git que apoya desde "la administración del proyecto hasta la integración continua" utilizando herramientas de terceros como Travis CI.

2.6.4. Heroku

"Heroku es un plataforma multi-lenguaje para aplicaciones en la nube que permite a los desarrolladores desplegar, escalar y manejar sus aplicaciones sin la necesidad de servidores o administración. Actualmente permite la utilización de lenguajes como Ruby, Java, Node.js, Python, PHP y Scala", fundada en 2007 por James Lindenbaum y Adam Wiggins en San Francisco, California en los Estados Unidos fue adquirida en 2010 por Salesforce. [32, 33, 34]

2.6.5. PostgreSQL

PostgreSQL o Postgres es un manejador de base de datos. Con "raíces en investigación universitaria PostrgreSQL no pudo haber alcanzado su éxito sin el Internet", debido a que este permitió a la comunidad mejorar y dar soporte al software que compite con las opciones comerciales. Actualmente cuenta con "más de 15 años de desarrollo activo". [35]

La implementación SQL de PostgresSQL cumple "con el estándar ANSI-SQL:2008", además cumple completamente con "las propiedades ACID". [36]

2.6.6. Express.js

Express.js o Express "es una infraestructura de aplicaciones web Node.js mínima y flexible que proporciona un conjunto sólido de características para las aplicaciones web y móviles". Cuenta con métodos HTTP y middleware para la creación de un API REST. [37]

Utilizado por empresas como Netflix, fue adquirido en 2014 por Strongloop para posteriormente ser adquirida por IBM, finalmente fue puesta bajo el cuidado de la Fundación Node.js en el año 2015. [38]

2.6.7. Node Package Manager

Un sistema manejador de paquetes o manejador de paquetes es una herramienta de software que realiza el proceso de instalación, actualización, configuración y remoción de programas de computo de manera consistente de un sistema operativo de una manera consistente.

Los sistemas manejadores de paquetes están diseñados para ahorrar tiempo y dinero a las organizaciones mediante la distribución de los paquetes software que elimina la necesidad de instalaciones y actualizaciones manuales.

Node Package Manager (npm) es el manejador de paquetes de Node.js, desarrollado en 2009 como un proyecto de código abierto "para ayudar a los desarrolladores JavaScript a compartir facilmente módulos de código empaquetados", actualmente cuenta con más de 250,000 paquetes registrados en el npm Registry, "una colección pública de paquetes de código abierto para Node.js". [39]

2.6.8. Gulp

Gulp es una herramienta que "ayuda a automatizar tareas en el proceso de desarrollo" [40]. Existen integraciones de Gulp "en todos los IDEs (Integrated Development Envirorment) más populares" y se utiliza en plataformas como PHP, .NET, Node.js y Java. Gulp utiliza modulos npm para realizar "más de dos mil" transformaciones mediante un API pequeño.

Algunos ejemplos de la utilización de Gulp es la compilación de archivos CoffeeScript a JavaScript, compilación de archivos SASS o SCSS a CSS y la creación servidores estáticos para pruebas.

2.6.9. JavaScript Standard Style

JavaScript Standard Style (Estilo de JavaScript Estándar) es un estándar web no afiliado con ningún grupo oficial que permite "mantener un alto estándar de calidad en nuestro código" [41]y permitir a nuevos contribuidores "seguir algunos estándares básicos de estilo". Las reglas del JavaScript Standard Style definen que se deberán utilizar dos espacios en las indentaciones; comillas simples en los textos; no tener variables no utilizadas; no utilizar puntos y coma; no inicar una línea con paréntesis; tener un espacio después de las condiciones y antes del primer paréntesis; paréntesis antes del nombre de las funciones y antes de los paréntesis; vigilar el manejo de los errores de JavaScript; entre otros.

2.6.10. JSdoc

JSDoc [42, 43, 44]en su versión 3 es un "API generadora de documentación para JavaScript". Funciona mediante los comentarios en el código de la aplicación que contienen descripciones del mismo en formato JSDoc, posteriormente se analizan todos los documentos de código especificados por el desarrollador y genera un sitio web con la documentación de la aplicación en formato HTML (Hyper Text Markup Language). La sintaxis de JSDoc es similar a la utilizada en Javadoc y a PHPDoc. "El propósito de JSDoc es documentar el API de tu aplicación Javascript o librería".

Los comentarios JSDoc se recomienda sean colocados "justo antes del código a documentar". La forma más simple de documentación es la descripción del código posterior al comentario. "Tags especializados para la documentación" pueden utilizarse para dar más información sobre el código, por

ejemplo, para especificar un constructor.

2.6.11. Crypto

Crypto es un modulo de Node.js que provee funcionalidades criptográficas para funciones de hash, HMAC, encriptación, desencriptación, firma y verificación. Actualmente el modulo se encuentra, de acuerdo a la clasificación de Node.js, en un estado estable. Además, existe una implementación parcial de este modulo para navegadores llamada crypto-browserify. [45, 46]

2.6.12. Browserify

Browserify, lanzada en 2011 y desarrollada por James Halliday, es una herramienta JavaScript de código abierto que permite a los desarrolladores escribir aplicaciones similares a Node.js y que son compiladas para poder ser utilizadas en el navegador.

2.6.13. FileSystem API

El FileSytem API [47, 48]es un API "no estándarizada y no en vías de estandarización" que "simula un sistema de archivos local con el que las aplicaciones web pueden trabajar" mediante la lectura, escritura y creación de archivos y directorios en ambientes cerrados.

Actualmente, FileSystem API solamente se encuentra disponible en el navegador Google Chrome desde la versión 13 y cuenta con dos modalidades: asíncrona, que es una función no bloqueante (puede ejecutar otros procesos al mismo tiempo); y síncrona, que regresa los valores deseados y no permite realizar ningun otro tipo de proceso en paralelo.