Extended software lifecycle management

Door: **De IT-Jurist**
Versie: 1.0
Datum: mei 2015

1. Inleiding

De IT-sector is in enkele tientallen jaren enorm gegroeid. Een van de redenen hiervoor is de toenemende inzet van software bij de ondersteuning van allerlei bedrijfsprocessen en de aansturing van apparatuur en machines. Organisaties verwachten namelijk dat automatisering hen het economische voordeel van kostenbesparing en het kwalitatieve voordeel van efficiëntere informatievoorziening en -verwerking brengt. Software en de daarmee verwerkte gegevens worden daardoor steeds essentiëler voor de continuïteit van de bedrijfsvoering voor de softwaregebruiker (in dit artikel is dat de software gebruikende organisatie). De softwaregebruiker is – hoewel hij daarvan niet altijd op de hoogte is - in beginsel verantwoordelijk voor het verzorgen van die continuïteit.

Een softwarepakket kan een levensloop hebben van tientallen jaren. Gedurende die levensloop wordt de software ontwikkeld en verhandeld door de leverancier, gebruikt door de softwaregebruiker en uiteindelijk uit gefaseerd. Het verloop van de levensloop van software kan op verschillende manieren invloed hebben op de continuïteitspositie van de softwaregebruiker. Hij is echter niet altijd op de hoogte van de gebeurtenissen binnen die levensloop. Hierdoor is het voor de softwaregebruiker niet controleerbaar of hij zijn verantwoordelijkheid voor het zorgen van zijn continuïteit volledig nakomt. De risico's die hiermee gepaard gaan, worden in deze tekst besproken.

2. Extended Software Lifecycle Management: een uitgebreidere kijk op de levensloop van software

In de softwarewereld is het begrip "software lifecycle" of "application lifecycle" niet onbekend. De term ziet voornamelijk op het (beheer van het) technisch proces van ontwikkeling en onderhoud van de software door de softwareleverancier. In dit artikel echter, wordt de levensloop van software in een breder perspectief geplaatst.

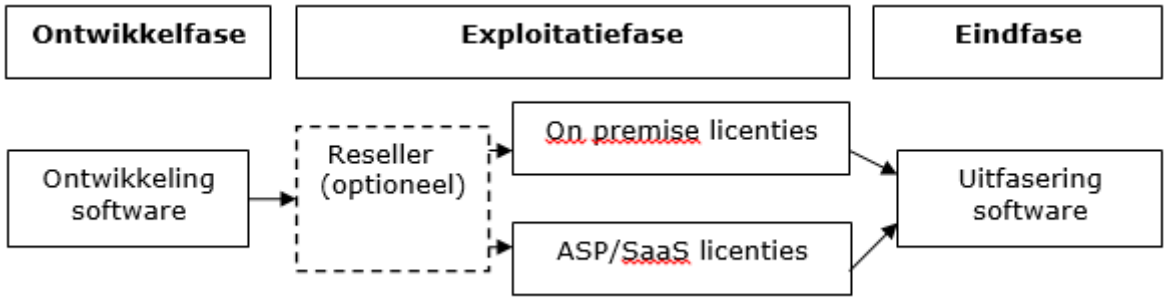
De levensloop van software is grofweg op te delen in een drietal fasen. In de ontwikkelfase wordt de eerste versie van de software tot stand gebracht door de (ontwikkelaars van de) softwareleverancier en er ontstaat een broncode. In deze fase ontstaat op grond van de Auteurswet voorts automatisch, dus zonder dat enige registratiehandeling is vereist, het auteursrecht op die software. De softwareleverancier is doorgaans de houder van dit auteursrecht. Het auteursrecht is het exclusieve recht op openbaarmaking en verveelvoudiging van de programmatuur, en rust op zowel de broncode als de objectcode van de software. De houder van het auteursrecht beslist daarom wat er met de software gebeurt en of, en zo ja hoe, deze wordt verhandeld. Hierdoor heeft software, naast het feit dat het een technisch goed is, een sterk juridisch karakter. In de ontwikkelingsfase zijn alle investeringen en inspanningen van de softwareleverancier gericht op het tot stand brengen van de software, aangezien daarmee omzet moet worden gegenereerd. Hierdoor heeft software ook het karakter van een asset, dus een goed met bedrijfseconomische waarde.

Nadat de software is ontwikkeld, wordt deze, met de mogelijke inzet van door hem ingeschakelde resellers, in de exploitatiefase verhandeld door de softwareleverancier en gebruikt door de softwaregebruiker. De exploitatie vindt plaats volgens a. het 'traditionele' licentiemodel waarbij de software 'on premise' op de apparatuur van de softwaregebruiker wordt geïnstalleerd of b. het ASP/SaaS-model waarbij de software en de daarmee verwerkte gegevens op afstand worden gehost in een door de leverancier beheerde omgeving. In het eerste geval krijgt de softwaregebruiker de beschikkingsmacht over een objectcodeversie van de software, in het tweede geval niet. De exploitatiefase beslaat (meestal) het grootste deel van een softwarelevensloop. In deze fase vinden, als gevolg van het handelen en

nalaten van de softwareleverancier, veel verschillende gebeurtenissen in de levensloop plaats.

De eindfase van de softwarelevensloop ten slotte, is aangebroken wanneer de software wordt uit gefaseerd door de softwareleverancier. Dit betekent dat de softwareleverancier, om redenen die van geval tot geval verschillen, de ondersteuning, het onderhoud en de verdere ontwikkeling van de software stillegt en het pakket uit de handel neemt.

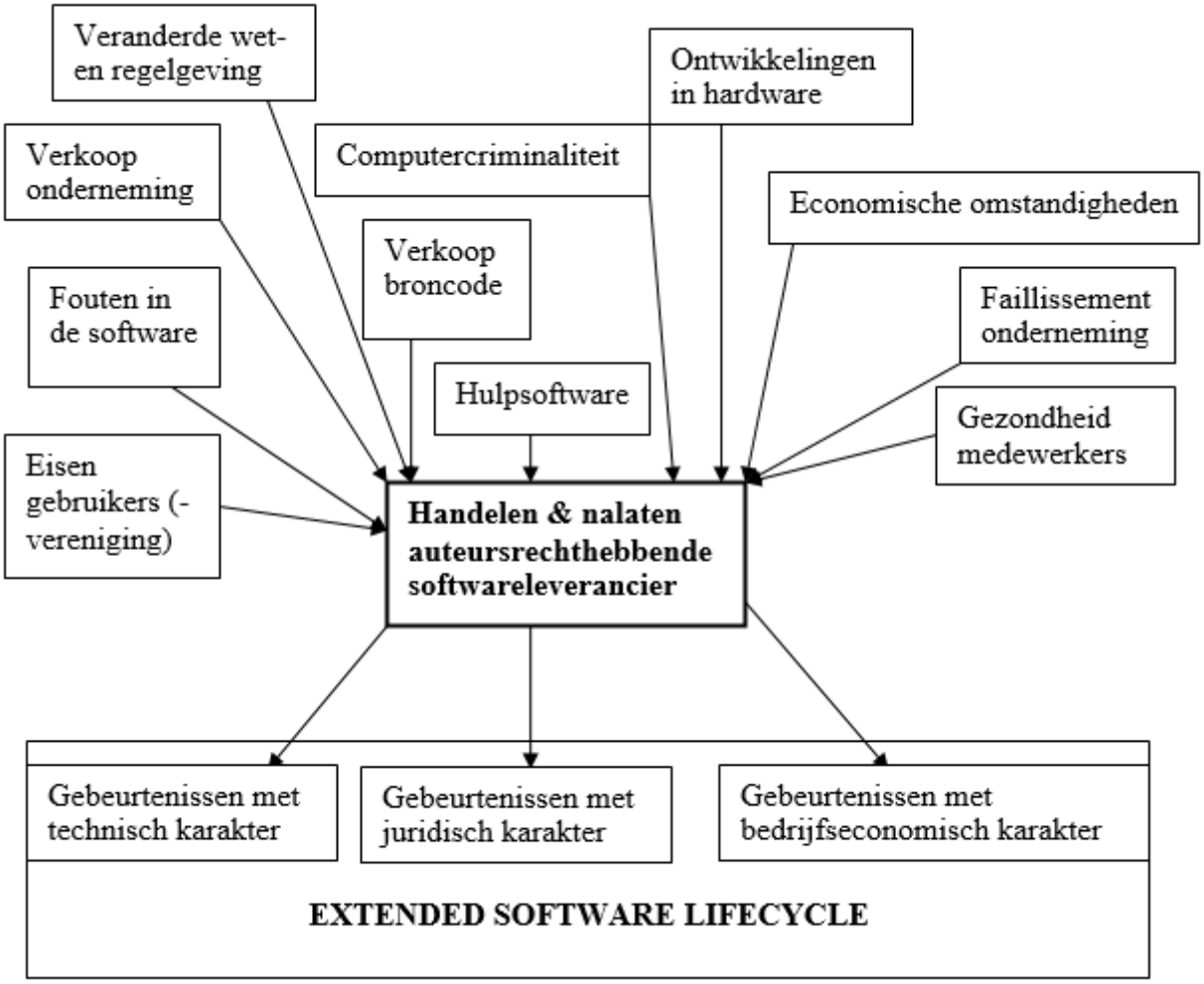
Schematisch als volgt voorgesteld:



Voor de softwaregebruiker is het belangrijk om zich te realiseren, dat het onderscheid tussen de hiervoor beschreven fasen in de levensloop van software in de praktijk niet zo scherp is als dat in dit artikel wordt neergezet. Er is in praktijk vaak sprake van overlap.

De gebeurtenissen in de levensloop van software doen zich, met uitzondering van het ontstaan van het auteursrecht, niet uit zichzelf voor. Zij zijn het rechtstreekse gevolg van handelen en nalaten dat door de softwareleverancier wordt geïnitieerd. Dat is logisch, aangezien hij in de regel de houder is van het auteursrecht op software. Hij heeft dus de noodzakelijke beschikkingsmacht. Ook beschikt de softwareleverancier over de technische, personele en financiële middelen in huis om de gebeurtenis te bewerkstelligen. Hij is daar, omdat aanleiding van een verandering in de levensloop van software buiten zijn invloed kan staan, vaak zelfs toe genoodzaakt: zo moet software in beginsel continu worden onderhouden. Uit de praktijk blijkt dan ook, dat het niet zo is dat een softwareleverancier achterover leunt op het moment dat de exploitatiefase is aangebroken. Software is daarom een dynamisch goed in een dynamische omgeving.

Schematisch als volgt voorgesteld:



Met een gebeurtenis in de levensloop van software betreft een verandering of juist het voortbestaan van een technische, juridische en/of bedrijfseconomische eigenschap van de software zelf. Daarom hebben ook de gebeurtenissen in de levensloop van software een technisch, juridisch en/of bedrijfseconomisch karakter. Die levensloop van elk softwarepakket ten slotte, is uniek.

3. Twee casussen

Om het voorgaande nader te duiden, wordt in deze paragraaf aan de hand van twee casussen geprobeerd om de (mogelijke) levensloop van een softwarepakket te visualiseren.

Casus 1 – Software voor metaalwarenfabriek
Een metaalwarenfabriek (de softwaregebruiker) wil software voor het uitvoeren van de productadministratie. De metaalwarenfabriek geeft de opdracht tot het ontwerpen en het ontwikkelen van de software aan één man (de leverancier). De metaalwarenfabriek krijgt een gebruikslicentie op de software, maar niet de toegang tot de broncode.

De leverancier voert ook het onderhoud aan de software uit. Hij is echter niet erg consequent met het documenteren van zijn handelingen. Na enkele jaren overlijdt de man die de

software heeft geleverd aan de metaalwarenfabriek. Het auteursrecht op de software gaat daarom over op zijn erfgenamen.

Omdat deze niets met de broncode van de software kunnen, verkopen de erfgenamen het auteursrecht en de software aan de metaalwarenfabriek, die de software wil laten onderhouden door een softwarehouse. Omdat vanwege de slechte documentatie rond de broncode het onderhoud niet goed door een ander dan de oorspronkelijke leverancier kan worden uitgevoerd, veroudert de software snel. Hierdoor is de metaalwarenfabriek gedwongen om over te stappen op andere programmatuur.

In deze casus zijn verschillende gebeurtenissen in de levensloop van software beschreven:

	Ontwikkelingsfase software	**Exploitatiefase software**	**Eindfase software**
Technische gebeurtenissen	- Ontwikkeling software	- Onderhoud software - Gebrek aan onderhoud software	
Juridische gebeurtenissen	- Ontstaan auteursrecht op software	- Vestiging gebruiksrecht op software - Overgang auteursrecht op software naar erfgenamen - Verkoop software aan metaalwarenfabriek	
Bedrijfseconomische gebeurtenissen		- Software veroudert	- Software niet meer ondersteund

Casus 2 – Software voor huisartsen
Een softwareleverancier maakt standaardsoftware waarmee huisartsen (de softwaregebruikers) patiëntgegevens kunnen bijhouden. De software wordt geïnstalleerd op de computerapparatuur van de huisarts, die een gebruiksrecht krijgt. De software wordt actief onderhouden door de softwareleverancier.

Na enkele jaren besluit de softwareleverancier tot het ombouwen van de software naar een SaaS-oplossing, omdat hij verwacht dat hierdoor zijn klantenbestand zal groeien. De werkzaamheden vergen echter een behoorlijke investering, waardoor de leverancier voor een lening aanklopt bij de bank. De bank verstrekt de noodzakelijke lening en krijgt als tegenprestatie de helft van het auteursrecht op de software.

Nadat de huisartsen zijn overgestapt op de SaaS-oplossing, worden de normen rond informatiebeveiliging in de zorg aangescherpt. De huisartsen willen voor de continuïteit van het onderhoud en het gebruik van de software en de beschikbaarheid van de met de software verwerkte gegevens een software escrowregeling. De softwareleverancier regelt dit, maar vraagt de bank niet om toestemming.

Omdat zijn twee beste ontwikkelaars bij de softwareleverancier vertrekken, vermindert de kwaliteit van de software. Hierdoor loopt het aantal gebruikers terug, waardoor de leverancier vanwege verminderde inkomsten in zwaar weer komt en uiteindelijk in staat van faillissement wordt gesteld. De huisartsen die de software nog wel gebruiken, doen een beroep op hun rechten uit de software escrowregeling. De bank, immers gedeeltelijk auteursrechthebbende, blokkeert echter de afgifte van de broncode aan de huisartsen.

Ook hiervan de gebeurtenissen in een overzicht:

	Ontwikkelingsfase software	**Exploitatiefase software**	**Eindfase software**
Technische gebeurtenissen	- Ontwikkeling software	- Onderhoud software	
Juridische gebeurtenissen	- Ontstaan auteursrecht op software	- Vestiging gebruiksrecht op software - Verdeling auteursrecht op software - Software opgenomen in escrowregeling - Software onderdeel faillissementsboedel	
Bedrijfseconomische gebeurtenissen		- Transformatie software naar SaaS-oplossing - Vermindering kwaliteit software	

Een schematische weergave van de levensloopgebeurtenissen van een softwarepakket schept verheldering en brengt overzicht. Tegelijkertijd zal men zich moeten realiseren dat de werkelijkheid complexer is en er overlap in tijd tussen de gebeurtenissen aanwezig kan zijn: het vestigen van een gebruiksrecht op de software bijvoorbeeld, kan plaatsvinden gedurende de periode dat er onderhoud wordt uitgevoerd. Daarnaast is er overlap in het karakter van de gebeurtenissen binnen een softwarelevensloop. Zo is het transformeren van software naar een SaaS-oplossing in bovenstaand schema als een bedrijfseconomische gebeurtenis aangemerkt, terwijl deze gebeurtenis echter ook een herinrichting van de techniek van de software betreft.

4. Bedreigingen voor de softwaregebruiker

Terug naar de softwaregebruiker, die moet zorgen voor zijn continuïteit. Ook voor hem is een softwarepakket een goed met een technisch, juridisch en bedrijfseconomisch karakter. De gebruiker krijgt de beschikking over, of in ieder geval de toegang tot , een stukje techniek, namelijk de objectcode van de software. Daarnaast krijgt hij een gebruiksrecht, dat hem in staat stelt om dat objectcode-exemplaar legaal te gebruiken. Ten slotte is software voor de gebruiker een (bedrijfseconomisch) productiemiddel dat hem ondersteunt in zijn bedrijfsvoering of deze zelfs voor hem uitvoert wanneer deze software bijvoorbeeld machines aanstuurt.

Software heeft een grote bedrijfseconomische waarde voor de softwaregebruiker wanneer deze bedrijfs-kritisch is. Dat is het geval wanneer de continuïteit van de bedrijfsvoering van de softwaregebruiker (deels) afhankelijk is van de continuïteit van de beschikbaarheid en het gebruik van de software en de daarmee verwerkte gegevens. De software uit beide casus in de vorige paragraaf zijn voorbeelden van bedrijfs-kritische programmatuur.

De grote bedrijfseconomische waarde van bedrijfs-kritische software voor de gebruiker, staat in schril contrast met de technische en juridische mogelijkheden die een gebruiker heeft om aan die waarde recht te doen. Met slechts de objectcode van de software kan hij de software bijvoorbeeld niet onderhouden of aanpassen aan de eisen die hij daaraan op grond van wet- en regelgeving moet stellen. Hij heeft dus niet de technische middelen in huis om iets met de software te kunnen doen. Het beperkte gebruiksrecht dat hij heeft, biedt hem die mogelijkheid ook niet.

Door een gebrek aan beschikkingsmacht, heeft de softwaregebruiker weinig invloed op de levensloop van een softwareproduct. De softwaregebruiker is daarvoor afhankelijk van het handelen en nalaten van zijn leverancier, de partij die deze beschikkingsmacht wel heeft. Een bekend voorbeeld is de afhankelijkheid van de softwareleverancier voor het onderhoud van de software: de gebruiker is daar zelf niet toe in staat.

De gebeurtenissen in de levensloop van software echter niet allemaal bekend voor de gebruiker. Hij kan bijvoorbeeld niet altijd (zelfstandig) constateren of het ontwikkelingstraject van software goed is gedocumenteerd en of het onderhoud van de software volledig wordt uitgevoerd. Dat geldt ook voor de kwaliteit van het programmeerwerk van de softwareleverancier: is dat conform de richtlijnen en best practices die in de automatisering gangbaar zijn? Voorts heeft hij geen zicht op juridische gebeurtenissen als het vestigen van een pandrecht op of de (gedeeltelijke) overdracht van het auteursrecht van software. Dat kan ook gelden voor bedrijfseconomische gebeurtenis als de uitfasering van een softwareproduct: dat is immers een proces dat door de softwareleverancier kan zijn opgestart zonder dat de gebruikers daarvan op de hoogte zijn.

Gebeurtenissen in de levensloop van software kunnen de continuïteitspositie van de softwaregebruiker bedreigen. Ten eerste bestaat het risico dat hem de technische mogelijkheid om op de beoogde wijze van de software gebruik te kunnen (blijven) maken, wordt ontnomen. Uit de eerste casus in de vorige paragraaf leidt een slecht gedocumenteerd ontwikkelingsproces bijvoorbeeld tot de stopzetting van het onderhoud van de software.

Onzichtbare gebeurtenissen met technisch karakter in softwarelevensloop
De softwaregebruiker heeft niet de mogelijkheid om onder de motorkap van zijn softwarepakket een kijkje te nemen, laat staan om daar handelingen te verrichten. Hij heeft allereerst geen zicht op het ontwikkelingsproces. De gekozen ontwikkelmethode bijvoorbeeld, beïnvloedt de kwaliteit van de software. Zo ligt bij een ontwikkelmethode als Scrum de nadruk meer op de totstandbrenging van een werkend product, terwijl het goed documenteren en het goed inrichten van de juridische aspecten van de software van ondergeschikt belang zijn. Daarnaast heeft de gebruiker geen zicht of bij de ontwikkeling en het onderhoud van de software aan daarvoor opgestelde best practices als COBIT wordt voldaan.

Voor de gebruiker van een ASP- of SaaS-systeem geldt, dat de configuratie van dat systeem aan zijn zicht is onttrokken. Hoe ziet bijvoorbeeld de keten van dienstverleners die de het systeem draaiend houden eruit? Welke beveiligingsmaatregelen zijn er getroffen en voldoet deze aan de relevante standaarden?

Ten tweede loopt de softwaregebruiker het risico dat hij het recht verliest om op een legale manier gebruik te kunnen maken van de software. In de tweede casus in de vorige paragraaf komt dat risico aan het licht door de gedeeltelijke overdracht van het auteursrecht op de software, alvorens deze wordt opgenomen in een software escrowregeling. De bank uit de casus wordt daarmee mede-beschikkingsbevoegd: zij moet toestemming geven voor de software escrow. Omdat hij deze niet heeft verleend, heeft zij de mogelijkheid om de afgifte van de broncode aan, en het gebruik daarvan door de softwaregebruiker tegen te houden. Het beperkte gebruiksrecht van laatstgenoemde biedt hem geen soelaas.

De gebeurtenissen in de levensloop van software bedreigen – nogmaals - de continuïteitspositie van de gebruiker van bedrijfs-kritische software. De softwaregebruiker is immers bedrijfseconomisch gezien afhankelijk van de software, maar technisch en juridisch gezien kan en mag hij er maar weinig mee verrichten. Het is voor hem noodzaak om gedurende de exploitatiefase zoveel mogelijk informatie omtrent zijn continuïteitspositie te verwerven om vervolgens maatregelen op maat te treffen. Hierbij is de medewerking van de softwareleverancier noodzakelijk, welke door hem niet zonder meer zal worden verleend. Aan de andere kant heeft hij ook voordelen bij het verstrekken van informatie over en het borgen van de continuïteitspositie van de softwaregebruiker: het verbetert bijvoorbeeld de kwaliteit van zijn dienstverlening.

5. Oplossingen voor de softwaregebruiker

Vanwege zijn beperkte invloed op de levensloop van software, heeft de softwaregebruiker niet de mogelijkheid om zijn continuïteitspositie op eigen initiatief te verbeteren. Dit terwijl hij, vanuit het oogpunt van de verzorging van zijn bedrijfscontinuïteit, daarvoor wel verantwoordelijk is. Recht doen aan de grote bedrijfseconomische waarde die bedrijfs-kritische software voor hem heeft, bestaat uit de verkrijging van een grotere beschikkingsmacht over a. het technische en b. het juridische goed software. Met die beschikkingsmacht kan de softwaregebruiker zijn continuïteitspositie veiligstellen. De

genomen maatregelen moeten de continuïteitsrisico's wegnemen, wat betekent dat er zij moeten worden getroffen voordat door een calamiteit die risico's werkelijkheid worden.

In het kader van het verkrijgen van een grotere beschikkingsmacht over het technische karakter van de software, kan de softwaregebruiker bijvoorbeeld de software laten auditen door een IT-auditor, om zeker te zijn dat de software voldoet aan de normen op het gebied van softwareontwikkeling en informatiebeveiliging. De softwaregebruiker verkrijgt daarnaast een grotere beschikkingsmacht over het juridische karakter van de software, wanneer hij bijvoorbeeld in de contractsverhouding met zijn leverancier zich ervan vergewist dat aan het recht om de software te gebruiken en aan de toegang tot de software als technisch object geen afbreuk kan worden gedaan.

De onzichtbaarheid die de gebeurtenissen in de levensloop van software kunnen hebben, kan er echter voor zorgen dat de getroffen maatregelen schijnzekerheid bieden. Zo heeft de metaalwarenfabriek uit de eerste casus van paragraaf 3 niets aan de toegang tot de broncode van de software, omdat achteraf bleek dat deze door de slechte documentatie van ontwikkeling en onderhoud niet bruikbaar was. De huisartsen uit de tweede casus kunnen daarnaast ondanks deelname aan een software escrowregeling de toegang tot de broncode worden ontzegd, omdat de mede-auteursrechthebbende bank geen toestemming voor de escrow heeft verleend.

Om zijn continuïteitspositie veilig te stellen, zal de softwaregebruiker zoveel mogelijk kennis moeten nemen van de gehele levensloop van de software die hij gebruikt, zodat hij adequate maatregelen kan treffen. De unieke levensloop die een softwarepakket heeft, maakt dat die maatregelen in alle gevallen maatwerk zijn. De kennis van de levensloop van software maakt dat maatwerk mogelijk.

Probleem is dat deze kennis slechts kan worden verkregen met medewerking van de softwareleverancier, en deze niet de plicht heeft om deze te delen. Daarnaast is het feitelijke verzorgen van de continuïteitspositie een multidisciplinaire aangelegenheid, die specialistische vaardigheden vereist die de softwaregebruiker doorgaans zelf niet in huis heeft en die centrale coördinatie vergt. Ten slotte hebben softwareleverancier en gebruiker uiteenlopende belangen, al kunnen zoals gezegd beide partijen hun voordeel doen met het beheer van de softwarelevensloop.

Bovenstaande leidt tot de conclusie dat het verstevigen van de continuïteitspositie van de softwaregebruiker de betrokkenheid en begeleiding van een onafhankelijke Trusted Third Party (TTP) eist. Deze partij moet beschikken over de technische en juridische kennis die noodzakelijk is om de gebeurtenissen in de levensloop van software te herkennen en de betekenis daarvan voor de positie van de softwaregebruiker, maar ook de softwareleverancier, te kunnen beoordelen. De TTP dient allereerst voor de registratie van de gebeurtenissen in een softwarelevensloop te zorgen, zodat de softwaregebruiker zijn continuïteitspositie op enig moment in die levensloop kan beoordelen. Ook de softwareleverancier heeft hierbij baat: de registers van een TTP kan hij gebruiken om zijn positie tegenover derden te handhaven. Daarnaast moet hij ook (via zijn netwerk) toegang hebben tot de specialisten die de noodzakelijke maatregelen voor het veiligstellen van de continuïteitspositie van de softwaregebruiker kunnen treffen. Ten slotte moet de TTP in staat zijn om deze specialisten aan te sturen. De TTP verzorgt daarmee het extended software lifecycle management dat in dit artikel wordt voorgestaan.

6. Samenwerking De IT-jurist en de IT-notaris

In de markt is reeds een partij actief die geschikt is om als TTP tussen softwareleverancier en softwaregebruiker op te treden: de IT-notaris. Hij heeft een wettelijk vastgelegde onafhankelijkheidspositie en een breed registratietakenpakket. Een belangrijk deel van de diensten van de notaris is daarnaast reeds afgestemd op de levensloop van zowel de natuurlijke personen als de rechtspersonen die hij bedient. Het beheer van een softwarelevensloop sluit hierbij goed aan. De zaken die door de notaris in een door hem opgestelde akte worden vastgelegd, kunnen niet worden aangevochten en zijn standplaats kan niet failliet gaan. De IT-notaris werkt samen met De IT-jurist, zodat hij toegang heeft tot een netwerk van IT-juristen en IT-deskundigen, waaronder IT-auditors en informatiebeveiligers.