Kooperationsrichtlinien

(1) Dieses Dokument präzisiert die allgemeinen Richtlinien für unsere Zusammenarbeit, deren spezifische Optionen im *Entwicklungs-* oder *Wartungsauftrag* vereinbart werden, der sich auf dieses Dokument bezieht.

(2) Kooperationspartner sind einerseits *Rumma & Ko OÜ* als **Dienstleister** und andererseits der **Auftraggeber** als Nutzer eines *Gesamtsystems*, das die *Endbenutzer* als „ihren Lino“ verstehen.

Was ist eine Lino-Anwendung?

(3) Eine **Lino-Anwendung** läuft auf einem Internet-Server und kann von jedem Desktop- oder Notebook-Computer bedient werden.

(4) Lino-Anwendungen nutzen das **Lino-Framework** (im Folgenden **Framework**), das seinerseits eine Vielzahl von weiteren freien Softwaremodulen nutzt, unter anderem Django und ExtJS.

(5) Das **Framework** besteht aus dem **inneren** *Framework* und der **Extensions Library** (**XL**), einer Sammlung anwendungsübergreifender Module. Das *Framework* bietet die Infrastruktur, mit der **Grundfunktionalitäten** wie Benutzerarten, Zugriffsrechte, Ansichtsparameter, Arbeitsabläufe, Notifikationen, Berichte und Ausdrucke, mehrsprachige Benutzerschnittstelle und mehrsprachige Kontakte usw. konfigurierbar sind.

(6) Der Dienstleister veröffentlicht das *Framework* und die Anwendung als **Freie Software**. Die Anwendung kann dadurch potentiell auch durch andere Organisationen und/oder Entwickler genutzt bzw. verändert werden. Der *Auftraggeber* ist nicht an den *Dienstleister* gebunden und kann sowohl Entwicklung als auch Wartung jederzeit einem anderen Anbieter übertragen.

Systemkomponenten

(7) Das Gesamtsystem besteht aus *Hardware, Systemsoftware, Anwendungssoftware,* lokalen Konfigurationsdateien und *Dokumentation*.

(8) Die **Hardware** besteht aus dem **Server**, d.h. einem eigenständigen (normalerweise virtuellen) Rechner innerhalb oder außerhalb der Gebäude des Auftraggebers, den **Clients**, d.h. den Rechnern der Endbenutzer, sowie dem **Netzwerk**, das die Verbindung der beiden ermöglicht.

(9) Unter **Systemsoftware** verstehen wir alle Software oberhalb von Lino. Dazu gehören das Betriebssystem sowie Systemdienste wie Web-Server, Datenbank-Server, Mail-Server, File-Server, cron, logrotate, supervisor. Wir verwenden üblicherweise *Debian* als Betriebssystem, *Apache* als Web-Server, *MySQL* oder *PostgreSQL* als Datenbank-Server, *Postfix* als Mail-Server.

(10) Unter **Anwendungssoftware** verstehen wir die im *Entwicklungs-* oder *Wartungsauftrag* benannte Lino-Anwendung, die der *Dienstleister* entwickelt und als freie Software öffentlich zur Verfügung stellt.

(11) Eine **Instanz** ist, wenn eine konkrete Version der *Anwendungssoftware* auf einem konkreten *Server* läuft. Sie besteht aus einem Domainname, hinter dem ein Webserver läuft, der Datenbank sowie den lokalen Einstellungen und Konfigurationsdateien zur Einbindung der *Anwendungssoftware* in die *Systemsoftware*.

(12) Auf einem *Server* können eine oder mehrere *Instanzen* laufen.

(13) Für die Bereitstellung und Wartung der *Hardware, einer gesicherten Zugriffsmöglichkeit für den Dienstleister* auf den Server, sowie **Backups** aller lokalen Daten ist der *Auftraggeber* verantwortlich.

(14) Für die Einrichtung und Wartung der *Instanzen* auf dem *Server* ist der *Dienstleister* verantwortlich.

Lebenslauf eines Projektes

(15) Im Laufe der Jahre durchläuft unsere Zusammenarbeit verschiedene Phasen.

(16) Wir unterscheiden zwischen *Neuentwicklungsprojekten*, Vorproduktionsbetrieb und *Produktionsbetrieb*. Im *Produktionsbetrieb* können gegebenenfalls *Weiterentwicklungsprojekte* gestartet werden.

(17) Ziel eines **Neuentwicklungsprojektes** ist die Inbetriebnahme einer neuen *Instanz*, auf der die *Software* läuft und benutzt werden kann. Ein *Neuentwicklungsprojekt* wird durch einen **Entwicklungsauftrag** gestartet.

(18) Je nach Umständen kann als optionaler Zwischenschritt ein **Vorproduktionsbetrieb** eingefügt werden.

(19) Ziel des **Produktionsbetriebs** ist die Aufrechterhaltung eines zuverlässigen Funktionierens einer *Instanz*.

Schritt	**Rechnung**	**Endet durch**
Interview und Vorgespräche	Kostenlos	Entwicklungsauftrag
Interne Entwicklungsphase zur Vorbereitung des Prototyps.	Festpreis für *Einrichtung des Prototypen (zahlbar bei Abnahme)*	Abnahme des Prototypen
Gemeinsame Entwicklungsphase	*Stundenkontingent*	*Einigung über Jahresbeitrag*
Vorproduktionsbetrieb	Pauschaler Beitrag	Unterzeichnung *Wartungsauftrag*
Produktionsbetrieb	Stundenkontingent	Ende des Wartungsauftrags

Neuentwicklungsprojekte

(20) Bei einem *Neuentwicklungsprojekt* durchläuft unsere Zusammenarbeit üblicherweise folgende Phasen und Entwicklungsschritte. Die Dauer jeder Phase hängt von vereinbarten Terminen, Verfügbarkeit des *Koordinator* und den sich ergebenden Fragestellungen ab.

(21) Der Dienstleister trifft sich mit dem Auftraggeber zu einem **Interview**. Ziel des Interviews ist es, die Bedürfnisse zu analysieren und zu klären, ob eine Lino-Anwendung dabei helfen könnte und welche Funktionen sie idealerweise haben müsste.

(22) Der Dienstleister setzt einen **Entwicklungsauftrag** auf, in dem die beim Interview besprochenen Funktionen der zu entwickelnden Anwendung beschrieben sind sowie eine Schätzung der Entwicklungskosten.

(23) Wenn der *Auftraggeber* den *Entwicklungsauftrag bestätigt hat, beginnt der Dienstleister* mit dem **Einrichten des Prototypen**. Ein **Prototyp** ist eine *Instanz* mit fiktiven Demo-Daten, die als Arbeitsgrundlage für die *aktive Entwicklungsphase* dient. Der *Prototyp* kann entweder auf einem vom Dienstleister betriebenem *Server* installiert sein, oder direkt auf dem *Server*, auf dem die *Software* später im *Produktionsbetrieb* laufen wird.

(24) Das Einrichten des Prototypen wird auch als **interne Entwicklungsphase** bezeichnet. Gegen Ende dieser Phase präsentiert der Auftraggeber den Prototypen.

(25) Mit der **Abnahme des Prototypen** bestätigt der *Auftraggeber*, dass die **gemeinsame Entwicklungsphase** beginnen kann. Der *Dienstleister* schreibt eine Rechnung über Einrichtung des Prototypen und Ankauf von Stundenkredit für die Weiterentwicklung.

(26) In der **gemeinsamen Entwicklungsphase** wiederholen sich die folgenden Etappen zyklisch: *Baustellenbesichtigung* mit dem Koordinator, Bearbeitung der festgestellten Probleme, Entwicklung und Installation einer neue Version und Planung der nächsten *Baustellenbesichtigung*. Wir arbeiten während dieser Phase auf dem *Prototypen*, der sich ständig verändert und weiterentwickelt.

(27) Gegen Ende der gemeinsamen Entwicklungsphase ist die Anwendung so weit fertig, dass die Datenbank bei Aktualisierungen nicht mehr gelöscht sondern migriert wird. In dieser Phase werden die fiktiven Testdaten gelöscht, die Konfigurationsdaten angepasst, eventuell werden Daten importiert aus bestehenden Quellen. Der *Koordinator* organisiert Schulung der Endbenutzer und Erfassung der Stammdaten. Probleme, die sich bei der täglichen Arbeit ergeben, werden erfasst, analysiert und behoben.

(28) Mit der **Abnahme der Instanz** bestätigt der *Auftraggeber*, dass die Instanz in den Vorproduktions- oder Produktionsbetrieb wechseln kann.

Vorproduktionsbetrieb

(29) **Vorproduktionsbetrieb** ist eine günstige Form der Zusammenarbeit zum gemeinsamen Durchleben von „Kinderkrankheiten“ einer neuen Anwendung. Eine Instanz im *Vorproduktionsbetrieb* wird zu einem individuell vereinbarten pauschalen **Jahresbeitrag** gewartet und weiterentwickelt.

(30) Im Vorproduktionsbetrieb sind **Support, Analyse** und **Optimierungen** für den Auftraggeber kostenlos.

(31) Als **Analyse** bezeichnen wir die Erfassung eines Problems, Rückfragen des Dienstleisters zur Präzisierung einer Änderungsanfrage, die Suche nach Ursachen und das Erarbeiten von Lösungsvorschlägen.

(32) **Optimierungen** sind Änderungen der Software, für die sich ein eigenes Angebot nicht lohnt.

(33) Als **Support** bezeichnen wir die Beantwortung von Fragen zur Benutzung der Software

Produktionsbetrieb

(34) Als Instanz im **Produktionsbetrieb** bezeichnen wir eine Instanz, für die ein **Wartungsauftrag** besteht.

(35) Im *Wartungsauftrag* übernimmt der *Dienstleister* die technische Verantwortung für das zuverlässige Funktionieren der *Instanz*.

(36) Wartungsaufträge können formlos verlängert werden, indem der *Dienstleister* eine Rechnung für die nächste Wartungsperiode ausstellt. Mit der Zahlung der Rechnung erklärt der *Auftraggeber* sein Einverständnis zur Verlängerung.

Weiterentwicklungsprojekte

(37) Ein **Weiterentwicklungsprojekt** *ist ein Entwicklungsprojekt, das* parallel mit dem Wartungsauftrag auf einer Instanz im Produktionsbetrieb läuft.

(38) Die Kosten für *Analyse, Planung* und *Implementierung* eines Weiterentwicklungsproj-

ekts trägt der Auftraggeber.

(39) Als **Implementierung** bezeichnen wir die Änderungen im Quellcode, der Dokumentation und der lokalen Konfiguration, den Support, die eventuelle weiterführende Analyse und die Durchführung aller Einsätze, die zur Lösung eines Problems nötig sind.

Aufgaben des Auftraggebers

(40) Der *Auftraggeber* muss Personal zur Verfügung stellen oder beauftragen, um folgende Aufgaben zu übernehmen:

(41) Ein **Koordinator** zentralisiert den Informationsfluss zwischen *Entwickler* und *Endbenutzern*, klärt Rückfragen des Dienstleisters zu Anfragen des Auftraggebers, verwaltet Zugriffsrechte der Endbenutzer und organisiert für diese je nach Bedarf Support, Dokumentation und Schulung.

(42) Als **Endbenutzer** bezeichnen wir alle Personen, denen der Auftraggeber Zugriff auf die Instanz gewährt. Der Auftraggeber ist verantwortlich für das Vergeben und Zurücknehmen von Zugriffsrechten.

(43) Der **Systemverwalter** ist eine vom *Auftraggeber* zur Verfügung gestellte oder beauftragte Person, die für Bereitstellung und Wartung der *Hardware* sowie Backups aller lokalen Daten verantwortlich ist.

(44) Ein einziger Mitarbeiter des Auftraggebers kann mehrere Rollen zugleich ausüben.

(45) Alle Rollen können bei Bedarf durch einen externen Partner ausgeübt werden, den der Auftraggeber getrennt beauftragt.

(46) Der Auftraggeber meldet eventuelle Probleme dem Dienstleister, beantwortet Rückfragen des Dienstleisters und liefert ihm alle nötigen Informationen, die er zur Lösungsfindung braucht.

(47) Der Auftraggeber ist mindestens per E-Mail erreichbar, idealerweise auch telefonisch, über *instant messaging* oder Internet-Telefon

(48) Der *Systemverwalter* teilt dem *Dienstleister* alle Informationen über Änderungen an der Hardware mit.

Aufgaben des Dienstleisters

(49) Der Dienstleister ist prinzipiell werktags von 8 bis 18 Uhr telefonisch **erreichbar**. Garantierte Bereitschaft nur nach vorheriger Absprache.

(50) Der Dienstleister beantwortet jede Problemmeldung des Auftraggebers innerhalb einer entsprechend der Dringlichkeit des Problems vertretbaren Zeit.

(51) Der Dienstleister führt alle Einsätze auf dem Server gewissenhaft durch und ohne den Produktionsbetrieb zu gefährden.

(52) Der *Dienstleister* veröffentlicht und wartet **technische Dokumentation** über die *Anwendungssoftware* in englischer Sprache. Die *technische Dokumentation* hat als Zielgruppe andere Entwickler und ist auch Bestandteil der *Testsuite*. Umfang der technischen Dokumentation liegt im Ermessen des *Dienstleisters*.

(53) Der *Dienstleister* wirkt nach eigenem Ermessen bei der **allgemeinen Entwicklung des Frameworks** mit*. Dazu gehören* z.B. Entwicklung neuer Front-Ends, Verbesserungen der Entwicklerschnittstelle, der Methoden für *Qualitätssicherung*, *Deployment* und Dokumentation, sowie funktionale Änderungen von allgemeinem Interesse.

(54) Als **Qualitätssicherung** bezeichnen wir das Schreiben und Pflegen von automatisierten Testverfahren.

(55) Als **Deployment** bezeichnen wir die Veröffentlichung der Software mit dem Ziel, sie

auch für andere Entwickler verfügbar zu machen.

(56) Der *Dienstleister* veröffentlicht und wartet nach eigenem Ermessen anwendungsübergreifende Dokumentation für Endbenutzer von Lino-Anwendungen.

Zahlbar oder nicht?

(57) Der Auftraggeber zahlt nicht für **Kundenbetreuung.** Dazu gehören das Aufsetzen von Angeboten, Auskunft zum Fortschreiten einer geplanten Arbeit und Antwort auf Rückfragen zu einem Dienstleistungsbericht.

(58) Der Auftraggeber zahlt für **Support** auch dann, wenn diese nicht zu einer Änderung führen.

(59) Der Auftraggeber zahlt für alle **Änderungen, die er anfragt**. Dazu gehört jede Arbeit des Dienstleisters, die zur Beantwortung nötig ist, zum Beispiel Support, Rückfragen zur Planung, Diagnose und *Analyse*, Änderungen in Quellcode und Dokumentation, Planung und Durchführung von *Einsätzen* auf dem Server sowie das Aufsetzen von Berichten, Tests und Dokumentation über diese Arbeit.

(60) Der Auftraggeber zahlt nicht für die allgemeine Entwicklung des **Frameworks** und Änderungen in *technischer Dokumentation*.

(61) Wenn eine Anfrage am effizientesten zu implementieren ist durch eine Änderung im Framework, gilt sie deshalb nicht als allgemeine Entwicklung und ist dennoch zahlbar. Der Dienstleister entscheidet über die technische Implementierung einer angefragten Änderung.

(62) Die Behebung von **Fehlern im Framework** ist kostenlos, wenn der Fehler keine Nebenwirkung einer vom Auftraggeber angefragten Änderung ist. Der Dienstleister entscheidet, ob und wie der Fehler behoben wird.

(63) Schriftliche Vorschläge zu einer **Verbesserung im Framework** oder der technischen Dokumentation werden kostenlos bearbeitet. Der Dienstleister entscheidet, wie und mit welcher Dringlichkeit der Vorschlag verwendet wird.

(64) Sollte bei einem Einsatz durch **mangelnde Vorsicht** des Dienstleisters der Produktionsbetrieb ausfallen oder nennenswert gestört sein, arbeitet der Dienstleister auf eigene Kosten, bis der Produktionsbetrieb wieder hergestellt ist.

(65) Für nennenswerte **neue Features im Framework**, deren Entwicklungskosten nicht an einen bestimmten Auftraggeber fakturiert wurden und bei denen der Auftraggeber wählen kann, ob er sie nutzt oder nicht, kann der Dienstleister einen einmaligen Beitrag zu seinen Entwicklungskosten verlangen, bevor er das neue Feature in die Instanz des Auftraggebers aufnimmt.

Dienstleistungsberichte und Stundenkontingente

(66) Der *Dienstleister* schreibt monatliche **Dienstleistungsberichte** über die geleisteten Arbeiten. Je nach Aktivität des Projekts können Dienstleistungsberichte auch häufiger oder seltener verschickt werden.

(67) Der *Dienstleister* führt für den Auftraggeber ein **Stundenkontingent**, d.h. ein Konto, dessen Stand durch Ankauf von **Stundenkredit** erhöht und durch einen *Dienstleistungsbericht* verringert wird.

(68) Zahlbare Arbeiten des Dienstleisters werden durch einen Dienstleistungsbericht vom Stundenkontingent abgebucht.

(69) Sollte ein angekaufter Stundenkredit innerhalb eines Jahres nicht aufgebraucht werden, verfällt er prinzipiell, kann aber nach Einverständnis des *Dienstleisters* um maximal

ein weiteres Jahr verlängert werden.

(70) Wenn der Saldo des Stundenkontingents **auf unter 10 Stunden** fällt, schreibt der Dienstleister dem Auftraggeber ein Angebot zum Ankauf zusätzlichen Stundenkredits, um das Stundenkontingent zu füllen.

(71) Wenn der Stundenkredit **auf unter 1 Stunde** fällt, arbeitet der Dienstleister ausschließlich an kostenlosen Anfragen.

(72) Wenn der Auftraggeber nicht einverstanden ist, für eine in einem Dienstleistungsbericht erwähnte Arbeit zu zahlen, teilt er dies dem Dienstleister spätestens zwei Wochen nach Erhalt des Dienstleistungsberichts schriftlich mit.

Aktualisierungen einer Instanz

(73) Jede *Instanz* ist individuell auf die Bedürfnisse des Auftraggebers zugeschnitten und kann bei Änderungen der Bedürfnisse angepasst werden.

(74) Ein **Einsatz** ist ein vom *Dienstleister* durchgeführter Eingriff auf den *Server* des *Auftraggebers*, bei dem die Konfigurierung des Servers geändert und/oder die Anwendungssoftware aktualisiert wird.

(75) Bei einer **Aktualisierung** wird neuer Quellcode der *Anwendungssoftware* auf die *Instanz* geladen und aktiviert.

(76) Ein **Release** ist eine *Aktualisierung*, für die eine eigene Versionsnummer und eigene Releasenotizen erstellt und gewartet werden.

(77) Eine **einfache Aktualisierung** kann durchgeführt werden, um Probleme nach einem *Release* operativ zu beheben. Dabei wird keine neue Versionsnummer vergeben und keine neue Testphase durchgeführt, und die Releasenotizen der laufenden Version werden lediglich aktualisiert.

(78) Bei jeder *Aktualisierung* ist der *Dienstleister* verantwortlich für eine korrekte **Datenmigration**.

(79) Lino enthält eine umfangreiche **Testsuite** zum automatisierten Testen vieler Funktionen. Die Testsuite wird automatisiert vor jedem Release durchgeführt.

(80) Die *Testsuite* kann keine absolute Sicherheit vor Problemen nach einem *Einsatz* geben.

(81) Eine **Regression** ist, wenn etwas, das vor einer Aktualisierung funktionierte und nachher nicht mehr.

(82) Eine **Nebenwirkung** ist etwas, das nach einer Aktualisierung anders funktioniert als zuvor und deshalb potentiell Verwirrung oder Supportbedarf schafft.

(83) Wir bieten zwei unterschiedliche Verfahren, um mit dieser Art von Problemen umzugehen. Im einfachen Testverfahren sind **Flexibilität** und **Aufwand** prioritär, im erweiterten Testverfahren eher **Stabilität** und **Planbarkeit**.

Einfaches Testverfahren

(84) Der Vorteil des **einfachen Testverfahrens** besteht darin, dass es kostengünstig und flexibel ist.

(85) Bei diesem Verfahren werden neue Versionen nach internem Test durch den Dienstleister über Nacht aufgespielt.

(86) Am nächsten Arbeitstag vereinbaren die *Endbenutzer* vor dem Einsatz eine **Testphase**, die eine oder mehrere Stunden dauern kann.

(87) In dieser Zeit testen ausgewählte Benutzer ihren neuen Lino im laufenden Produkti-

onsbetrieb.

(88) Sie sind dabei auf *Regressionen* und Überraschungen gefasst, die sie dann melden müssen und die dann prioritär bearbeitet werden.

(89) Alle sind sich dessen bewusst, dass der Einsatz rückgängig gemacht und die Instanz auf den Zustand vom Vortag zurück gesetzt werden könnte, falls ein Release-Blocker gemeldet wird.

Erweitertes Testverfahren

(90) Beim **erweiterten Testverfahren** arbeiten wir mit einer *Vorschau* auf die kommende Version, die von ausgewählten Benutzern gründlich getestet wird, bevor sie in Produktion geht.

(91) Der Vorteil des *erweiterten Testverfahrens* besteht darin, dass es reibungslosen Produktionsbetrieb ohne Unterbrechung gewährleistet.

(92) Eine **Vorschau** ist eine zusätzliche Instanz neben der Produktionsinstanz, üblicherweise auf dem gleichen Server, der eine Kopie der Produktionsdaten zeigt, wie sie mit der nächsten Version der Software aussähen.

(93) Der *Dienstleister* schreibt bei jedem *Release* **Releasenotizen** in der vereinbarten Sprache (Deutsch, Französisch oder Englisch) mit detaillierten Testanweisungen für die Tester.

(94) Eine **Testphase** startet, wenn der Dienstleister die Releasenotizen veröffentlicht hat. Während der Testphase prüfen die Endbenutzer, ob die kommende Version funktioniert.
(95) Wenn eine *Testphase* begonnen hat, darf der Dienstleister keine weiteren ungefragten Änderungen an der Instanz durchführen.

(96) Die *Testphase* dauert so lange wie nötig. Sie endet, indem die *Vorschau* in den *Produktionsbetrieb* geschaltet wird. Der Auftraggeber entscheidet, wann das ist.

(97) Nach der *Testphase* wird die nächste *Testphase* in einer neuen Instanz vorbereitet. Wenn die Vorbereitungen beendet sind und die nächste Testphase beginnen kann, veröffentlicht der *Dienstleister* die neuen Releasenotizen.

Autorenrecht und Datenschutz

(98) Alle Mitarbeiter des Dienstleisters, die Zugriff auf den Server oder Kontakt mit Endbenutzern haben, unterliegen den Datenschutzbestimmungen, die der Auftraggeber an seine eigenen Mitarbeiter stellt.

(99) Der Auftraggeber ist dafür verantwortlich, den Dienstleister über seine Datenschutzbestimmungen zu informieren.

(100) Der Dienstleister ist dafür verantwortlich, seine Mitarbeiter an die Datenschutzbestimmungen des Auftraggebers zu binden.

(101) Der Dienstleister veröffentlicht den im Rahmen der Zusammenarbeit erstellen Quellcode und Dokumentation der Software unter einer freien Lizenz.

(102) **Betriebsabläufe** und **Vornamen von Mitarbeitern** gelten nicht als vertraulich und dürfen in Quellcode und Dokumentation veröffentlicht werden.

(103) **Lokale** Konfigurationsdateien auf dem Server gelten als vertraulich und dürfen nicht veröffentlicht werden.

Haftungsausschluss

(104) Der Dienstleister garantiert weder Fehlerfreiheit der Software noch Ausfallsicherheit der Instanz, sondern seine Hilfestellung bei Problemen.

(105) Sollte jemand aufgrund eines Fehlers oder Ausfalls zu Schaden kommen, kann der Dienstleister dafür nicht rechtlich haftbar gemacht werden.

Sonstiges

(106) Die Unterschrift eines Entwicklungs- oder Wartungsauftrags bestätigt das prinzipielle Einverständnis beider Parteien mit den Bedingungen unserer Zusammenarbeit. Die Verpflichtungen des Dienstleisters beginnen mit dem Eingang der Zahlung einer Rechnung und enden automatisch bei negativem Stundenkontingent.

(107) Die Zusammenarbeit ist automatisch **beendet**, wenn das *Stundenkontingent* des Auftraggebers auf Null fällt oder der *Vorproduktionsbetrieb* abläuft und keine Einigung über die Fortsetzung der Zusammenarbeit gefunden wird.

(108) Dieses Dokument ist öffentlich und gilt für alle Aufträge, die sich darauf beziehen.
Die jeweils aktuelle Version steht unter
https://saffre-rumma.net/dl/Kooperationsrichtlinien.pdf

(109) Der *Dienstleister* kann dieses Dokument jederzeit anpassen und informiert den *Auftraggeber* über wichtige Änderungen. Prinzipiell gilt die am Rechnungsdatum veröffentlichte Version.

(110) Vereinbarte Preise werden jedes Jahr entsprechend des belgischen **Gesundheitsindexes** angepasst.

(111) Eine E-Mail gilt als **schriftliche Mitteilung**, wenn ihr Eingang vom Empfänger bestätigt wurde.