asseco
CZECH REPUBLIC

Technické řešení č. 6

POPIS KOMUNIKACE S PORTÁLEM (KLIENT)

ŘEŠENÍ A SPRÁVA
INTERNETOVÉHO PORTÁLU

ZDRAVOTNÍCH POJIŠŤOVEN

Verze 4

Asseco Czech Republic, a.s.

Autor: Vladimír Beneš

Obsah

Seznam příloh

Název	Jméno dokumentu
Model tříd a definice typů	P4ZP_PHP-APP_KOM_TR06_P01.DOC
Definice struktury XML	portal_kl.dtd
Komunikace portálu ZP - vzory XML	Komunikace Klienta Portálu - vzory XML.zip
Datové rozhraní pro předávání žádosti o vyúčtování	P4ZP-PHP_APP_VYU_P01.doc
Datové rozhraní pro předání hromadného oznámení zaměstnavatele	P4ZP-PHP_APP_HOZ_P01.doc
Datové rozhraní pro předání přehledu plateb pojistného zaměstnavatele	P4ZP-PHP_APP_PPPZ_P01.doc
III. - Datové rozhraní VZP ČR - speciální rozhraní	DRspec.pdf
Datové rozhraní pro verifikaci zdravotnických zařízení	P4ZP-PHP_APP_VERZZ_P01.doc
Vyúčtování zdravotní péče - popis kontrol	P4ZP_PHP-APP_VYU_TR08.DOC
Přesměrování zabezpečeným kanálem	P4ZP-PHP_APP_PRS_P01.doc

1. Obecné

Tento dokument **upřesňuje** CASE diagram (umístěný v datastore p4zp - model PHP-aplikace):

- UXML-XML struktury pro komunikaci s klienty (CLD)

Výstupy z tohoto modelu jsou **nezbytnou přílohou** tohoto dokumentu. Aktuální stav dané části modelu je umístěn v souboru **P4ZP_PHP-APP_KOM_TR06_P01.DOC**.

Jedná se o struktury pro definici XML formátu v komunikaci mezi portálem a klientem, čili o podklad pro DTD.

Pokud není uvedeno jinak, komunikace je realizována prostřednictvím protokolu HTTPS - **anonymního, tzn. klient nemusí použít k navazování HTTPS spojení svůj osobní certifikát**[1]. Iniciující strana vytvoří HTTP POST požadavek, ve kterém předá parametr *request* obsahující XML data pro výzvu.
Parametr *request* ale nemusí obsahovat přímo vlastní data. Pomocí tohoto parametru lze předat soubor obsahující XML data pro výzvu. Tento soubor může být komprimovaný - buď kompresí ZIP (soubor pak musí mít příponu .zip) nebo GZIP (a přípona souboru pak musí být .gz).
Další možností, jak předat výzvu v komprimované podobě, je předání komprimovaných dat (kódování base64) v parametru *request* a dále post parametru *request_compression* s hodnotou dle typu použité komprese „zip“ či „gz“.
Je lhostejné, jakým způsobem je parametr *request* plněn (zda [ne]komprimovaný soubor či komprimovaná data s příznakem v *request_compression* či přímo nekomprimovaná data) a na další zpracování toto nemá vliv. Komprimovaný soubor lze s výhodou použít u rozsáhlé výzvy.
Odpovědí je opět XML dokument.
Není-li uvedeno jinak, komunikaci zahajuje klient a portál odpovídá.
Adresa komunikačních bran portálu pro jednotlivé ZP je **https://<hostname>/kom_brana.phtml** (kde <hostname> je jméno domény portálu příslušné ZP – viz např. v přihlašovací stránce) a DTD definice XML, kterou musí veškerá komunikace splňovat, je ve všech případech ve společném souboru uloženém v http://www.portalzp.cz/portal_kl.dtd

Pro XML dokument výzvy či odpovědi platí následující pravidla (mj. vyplývající z výše uvedených CASE diagramů):

- Splňuje formální pravidla pro XML
- Odpovídá definici příslušného DTD souboru
- Hlavní element (**Komunikace**) obsahuje právě 2 základní elementy:
  - **Data** - vlastní předávaná data. Kromě předávaných dat, která závisí na konkrétní komunikaci (rozepsáno dále) obsahuje tyto atributy:
    - **Typ** - směr komunikace (POZADAVEK pro její zahájení (HTTPS request) a ODPOVED pro její ukončení)
    - **Ucel** - funkce, pro kterou klient předává podání – rozepsáno dále. Odpověď na požadavek komunikace konkrétního účelu má stejný účel, liší se tedy jen v atributu **Typ** a samozřejmě i v obsahu elementu **Data**.
    - **PZP_IdPodani** - má smysl jen při odpovědi portálu. Vrací zde jednoznačnou identifikaci založeného podání. I v případě, že došlo k chybám a podání nebude předáno ZP, je na portálu uloženo - jen z důvodu případné pomoci při odstranění problémů.
    - **PZP_Chyba** - má smysl jen při odpovědi portálu. Pokud je zde nula, podání bylo uloženo bez chyb a bude předáno ZP. V ostatních případech nebude předáno ZP, je zde předán kód chyby a v XML elementu Soubor je uveden podrobnější popis chyby.
  - **Podpis** - elektronický podpis elementu data včetně jeho počátečního a koncového tagu. Podepsáno certifikátem, který je uložen jak na Portálu tak u klienta.

POZOR: Vzhledem k tomu, že k podepisování lze použít i kvalifikovaný osobní certifikát a vzhledem k tomu, že od 17.08.2007 platí právní norma ČSN ETSI TS 101 456, autentizace probíhá pouze na

[1] V předchozí verzi komunikace se jednalo o HTTPS neanonymní, kde se klient autentizoval certifikátem. Od 17.08.2007 tato autentizace není legitimní (kolize s ČSN ETSI TS 101 456). Autentizace nově probíhá na základě rozboru HTTP POST parametru *request*, tzn. tento parametr musí splňovat určité kritéria (viz dál). Podrobnosti jsou na http://www.micr.cz/scripts/detail.php?id=3533

základě dohledání certifikátu použitého pro podpis, tzn. podepisovaný element ***Data*** *by měl z hlediska autentizace mít takový obsah, aby této normě vyhovoval, tzn. musí obsahovat srozumitelnou deklaraci, co klient podpisem zamýšlí (přihlašuje se do Portálu konkrétní ZP, současně předává podání konkrétní agendy v definovaném rozhraní a po předání podání + stažení protokolu se z portálu odhlašuje) (řeší to zanořený element* ***Prihlaseni****). Navíc musí klient celému podání rozumět, tzn. dále předávané soubory by měly být předávány jako soubory textové s transformací jen těch znaků, které by bez formátování byly v kolizi s definicí XML. Podepisovaný text (obsah elementu* ***Data****) musí klient vidět (****toto je nutno zajistit na úrovni klientské aplikace****).*
Aby se zabránilo podvržení teoreticky zachyceného podání třetí stranou, obsah elementu ***Data*** *musí být v rámci všech podání podepsaných daným certifikátem unikátní (server toto kontroluje). K tomuto účelu lze využít atribut* ***UnikatniInfo*** *v elementu* ***Prihlaseni****, přičemž i této informaci musí klient rozumět - lze využít např. aktuální datum + čas s přesností na sekundu.*

Vlastní obsah elementu **Data** je dále alespoň 1 element **Soubor**. Kódování souboru (atribut **Soubor.Format**) je volen jen pro vlastní přenos dat; na vlastní funkci nemá žádný vliv.
V případě směru komunikace „POZADAVEK“ (výzva klienta) se tímto způsobem připojují jednotlivé soubory daného druhu podání, přičemž jejich jméno (atribut **Jmeno**) a obsah (obsah elementu **Soubor**) musejí odpovídat konvencím dané funkce.
V případě směru komunikace „ODPOVED“ (odpověď portálu) se tímto způsobem předává buď protokol (pokud **Data.PZP_Chyba** je roven nule) nebo chybový soubor (v ostatních případech).

1.1 Podepisování XML

Podepisovaná je veškerá komunikace. Mechanismus je volen co nejjednodušší, stále stejný pro jakoukoli komunikaci, obecný pro jakkoli rozsáhlá data, normalizovaný a obsahující jen nezbytně nutné části.
Každá kumunikující strana musí mít uložený certifikát strany opačné. Způsob distribuce certifikátů je již mimo hranice tohoto podsystému.
XML dokumenty zahrnují element **Podpis**, jehož vlastní data obsahují podpis elementu **Data** (včetně počátečního a koncového tagu) **a nic víc** (tedy žádné dodatečné mezery, odřádkování apod).

Vlastní podpis je realizován formou PKCS#7. Data podpisu již nesmějí z kapacitních důvodů obsahovat podepisovaná data (přenášely by se v rámci stejného XML duplicitně). Neobsahují ani žádné certifikáty (obě komunikující strany musejí znát certifikáty stran opačné).

Jedním z možných běžně rozšířených softwarů pro práci s podpisy standardizovaných dle PKCS#7 je OpenSSL. Dokumentace k OpenSSL je na webu na http://www.openssl.org/docs/apps/openssl.html
Obecně http://www.openssl.org/ - odtud lze přejít mj. na dowload (GNU licence).

V dokumentaci tohoto Technického řešení č. 6 jsou pak uváděny příklady příkazů pro operace specifické pro práci s podpisy, tzn. jedná se o výsek dokumentace uvedené na webu. Tyto příklady pak vedou ke zjednodušení vývoje pro práci s podpisy (netřeba pak v rozsáhlé dokumentaci OpenSSL hledat způsob jeho použití pro tuto oblast - rovnou je zdůrazněno, na co je třeba se zaměřit a jsou uvedeny funkční příklady).

1.1.1 Tvorba podpisu

Podpis nezahrnuje ni certifikát, ni data. Splňuje formát PKCS#7 a může být tedy vytvořen pomocí utility OpenSSL např. příkazem:

```
openssl smime -sign -outform pem -signer signer.pem -in PodepisovanaData -out PodpisDat -nocerts
```

kde:

Funkce nebo výraz	**Popis**
openssl smime	Využití rozhraní pro tvorbu S/MIME mail utility OpenSSL.
-sign	Budou podepisována nějaká data
-outform pem	Výstup bude v textové podobě PEM, tzn. DER pod Base64
-signer signer.pem	Soubor obsahující podepisovací certifikát a privátní klíč (signer.pem)
-in PodepisovanaData	Soubor obsahující podepisovaná data (PodepisovanaData)

-out PodpisDat	Výstupní soubor podpisu (PodpisDat)
-nocerts	Nepřikládat certifikáty k podpisu

Pro úplnost doplňuji další užitečný parametr, který se zde ale nevyužije:

-nodetach	Vlastní podepisovaná data se pak začlení do výsledného podpisu (verifikace těchto dat: viz následující kapitola)

Více viz http://www.openssl.org/docs/apps/smime.html#

1.1.2 Verifikace podpisu

Opět pomocí OpenSSL lze podpis bez přiložených certifikátů a vlastních dat ověřit např. takto:

```
openssl smime -verify -noverify -inform PEM -in PodpisDat -out OverenaData -content PodepisovanaData -certfile cert.pem
```

kde:

Funkce nebo výraz	**Popis**
openssl smime	Využití rozhraní pro tvorbu S/MIME mail utility OpenSSL.
-verify	Bude ověřen podpis
-noverify	Nebude ověřen certifikát. Vzhledem k řešení není ověření certifikátu nutné. Vypuštěním tohoto parametru bude certifikát ověřován.
-CAfile CABundle	Má smysl jen při nepoužití parametru -novery. CABundle je soubor obsahující certifikáty důvěryhodných certifikačních autorit. Certifikát je pak ověřován proti nim.
-inform PEM	Vstupní data podpisu jsou ve formátu PEM, tzn. DER pod Base64
-in PodpisDat	Soubor obsahující podpis dat (PodpisDat)
-out PodepisovanaData	Výstupní soubor, do kterého utilita umístí ověřená data. Nebude-li tento parametr uveden, budou tato data vypsána na stdout.
-content PodepisovanaData	Soubor obsahující data, jejichž podpis se ověřuje (PodepisovanaData)
-certfile cert.pem	Certifikát podpisu. Obsahuje veřejný klíč párový ke klíči privátnímu v signer.pem uvedenému v předchozí kapitole.

Návratový kód: je-li 0, je podpis korektní. Stručná zpráva je dále vypsána na stdout (např. "Verification Successful").

Opět pro úplnost ověření podepsaných dat, která jsou zahrnuta do podpisu: postačí vypustit parametr "-content PodepisovanaData". Do souboru určeném parametrem "-out PodepisovanaData" je pak vylit obsah podepsaných dat.

Více viz http://www.openssl.org/docs/apps/smime.html#

Proč podepsaná data neobsahují vlastní data? Protože tato data jsou již v XML uvedena - XML by se jinak zhruba zdvojnásobilo co se velikosti týká).

Proč certifikát není uložen v podepsaných datech? Protože nelze použít jakýkoli certifikát, ale jen ten, který je zaregistrovaný na protější straně. Protější strana si pak sama aktivně musí zjistit, kterým momentálně přípustným certifikátem bylo podepisováno (jinak by při chybné implementaci třeba jen konstatovala, že jakýsi certifikát, který je obsahem podpisu, skutečně podepisuje daná data a data jsou tedy věrohodná).

2. Jednotlivé druhy (účely komunikací)

2.1 Předání vyúčtování zdravotní péče (Data.Ucel="VYU")

Ve výzvě klient předá soubor faktury (fdavka). Dále může předat soubor výkonů (kdavka). Pravidla pro pojmenování a obsahy těchto souborů jsou identická s pravidly v interaktivní aplikaci – viz přílohy **P4ZP-PHP_APP_VYU_P01.doc** a **P4ZP_PHP-APP_VYU_TR08.DOC**.

2.2 Předání registračních lístků (Data.Ucel="ORL")

Ve výzvě klient předá jen soubor výkonů (kdavka) obsahující výhradně dávky 80. Jedná se o alternativu ke komunikaci dle kap. 2.1 s tím, že se zde nepředává soubor faktur (fdavka), který u registračních lístků postrádá smysl.

2.3 Kontrola čísel pojištěnce pro SZZ (Data.Ucel="VERPOJ")

Ve výzvě klient s oprávněním na alespoň jedno zdravotnické zařízení předá seznam pojištěnců. V odpovědi obdrží info o přiřazení jednotlivých pojištěnců k pojišťovnám zúčastněných na projektu Portál ZP.
Aplikace má k dispozici jen údaje o pojištěncích těchto pojišťoven, proto nemůže verifikovat vztah dotazovaných pojištěnců vůči ostatním pojišťovnám. Pro libovolný den a libovolného pojištěnce tak lze určit, u které z pojišťoven zúčastněných na projektu Portál ZP byl pojištěnec pojištěn nebo určit, že odpověď je neznámá (tzn. buď u jiné pojišťovny nebo u žádné - formálně u žádné z důvodu dodržení rozhraní).
Popis rozhraní (názvy vstupního a výstupního souboru; struktura dat) je v příloze **DRspec.pdf** (kap. "III. - 3.2. Kontrola čísel pojištěnce pro SZZ"). Název vstupního souboru nemusí odpovídat dané metodice (pro tuto funkcionalitu je bezpředmětný), pozor na formát datumů.
Při této komunikaci se negeneruje žádné podání do pojišťovny.

2.4 Ověření zdravotnického zařízení (Data.Ucel="VERZZ")

Ve výzvě klient předá seznam zdravotnických zařízení. V odpovědi dostane obdobný seznam s informacemi o smluvnosti daných zdravotnických zařízení vůči aktuální zdravotní pojišťovně a pro známá zdravotnická zařízení i kontaktní údaje.
Popis rozhraní (názvy vstupního a výstupního souboru; struktura dat) je v příloze **P4ZP-PHP_APP_VERZZ_P01.doc**.

2.5 Předání hromadného oznámení zaměstnavatele (Data.Ucel="HOZ")

Ve výzvě klient předá soubor hromadného oznámení zaměstnavatele. Pravidla pro pojmenování a obsah tohoto souboru jsou identická s pravidly v interaktivní aplikaci (Hromadné oznámení zaměstnavatele – vstup ze souboru) – viz příloha **P4ZP-PHP_APP_HOZ_P01.doc**.

2.6 Předání Přehledu plateb pojistného zaměstnavatele (Data.Ucel="PPPZ")

Ve výzvě klient předá soubor přehledu plateb pojistného zaměstnavatele. Pravidla pro pojmenování a obsah tohoto souboru jsou identická s pravidly, dle kterého se v podepisovací komponentně vytváří příslušný soubor – viz příloha **P4ZP-PHP_APP_PPPZ_P01.doc**.

2.7 Přesměrování zabezpečeným kanálem (Data.Ucel="PRESMER")

Portál ZP zde slouží jako autentizační server. Přijme výzvu od klienta, otestuje, zda se jedná o validního klienta, zda výzva obsahuje všechny požadované náležitosti (a nic navíc), zda požadované náležitosti jsou syntakticky i sémanticky správně, a pokud ano, předá tuto výzvu (případně doplněnou dalšími informacemi o klientovi) zavedeným komunikačním kanálem do pojišťovny. Pokud i pojišťovna souhlasí, předá portál klientovi parametry na přesměrování ke vzdálené aplikaci. Tyto parametry následně klient použije pro přihlášení se k této aplikaci.

Viz schéma:

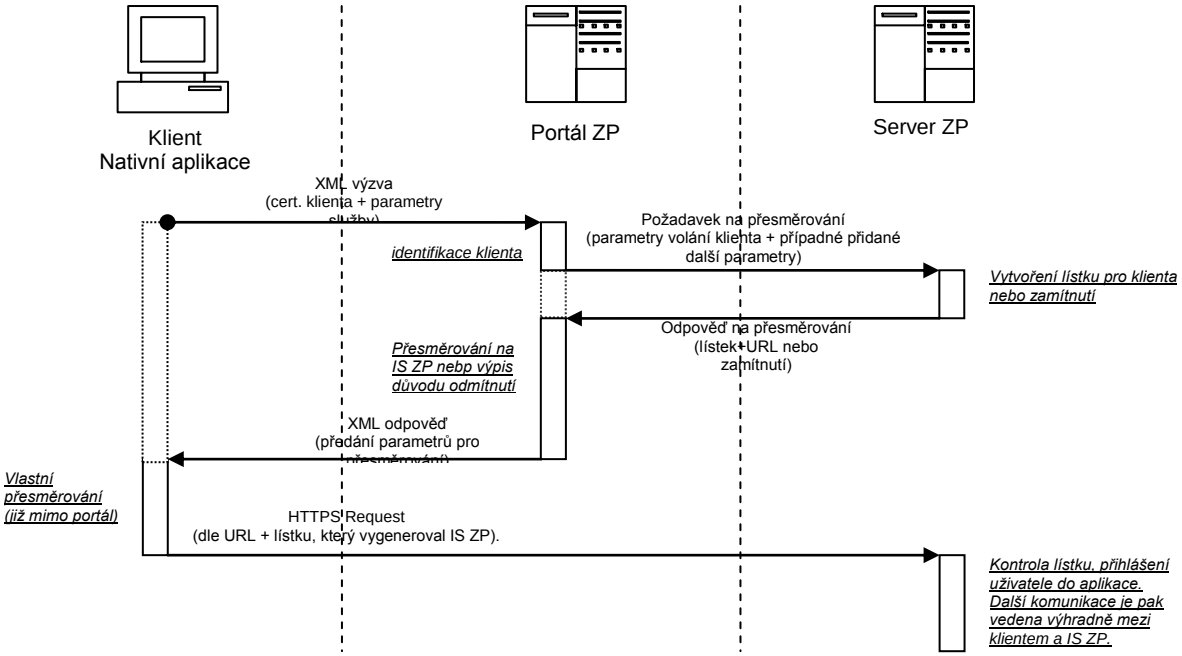

Schéma neobsahuje okamžité odmítnutí přesměrování ze strany portálu (bezprostředně po výzvě klienta) – např. z důvodu odmítnutí autentizace na portálu, syntaktické či sémantické chyby apod.

Komunikace mezi Portálem ZP a IS ZP je popsána v samostatné dokumentaci pro ZP a přesahuje rámec Technického řešení č. 6.

V rámci žádosti o přesměrování se předává soubor dle datového rozhraní definovaného v příloze - **PHP_APP_PRS_P01.doc**. Tamtéž je definováno i datové rozhraní pro akceptaci přesměrování. Při případném odmítnutí je vrácen html protokol se zevrubnějším popisem důvodu odmítnutí.