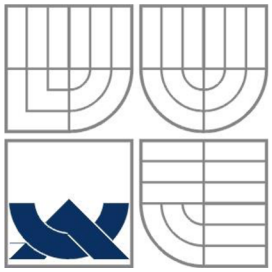
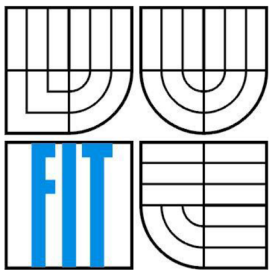


VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÝCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER SYSTEMS

PROCESNÍ PLATFORMA PRO INFORMAČNÍ SYSTÉM NÁRODNÍ DOTACE

PROCESS PLATFORM FOR THE NATIONAL SUBSIDIES INFORMATION SYSTEM

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE
AUTHOR

Bc. Martin Orava

VEDOUCÍ PRÁCE
SUPERVISOR

RNDr. Marek Rychlý, Ph.D

BRNO 2016

Zadání diplomové práce/18843/2015/xorava01

Vysoké učení technické v Brně - Fakulta informačních technologií

Ústav informačních systémů Akademický rok 2015/2016

Zadání diplomové práce

Řešitel: **Orava Martin, Bc.**
Obor: Informační systémy
Téma: **Procesní platforma pro informační systém Národní dotace**
Process Platform for the National Subsidies Information System
Kategorie: Informační systémy

Pokyny:

1. Seznamte se se požadavky na Informační systém Národní dotace 2016 a se specifikací BPM/ESB AgriBus. Zaměřte se zejména na integraci systému do ArgiBus. Prozkoumejte existující BPM/ESB systémy, které implementují podobné funkce, jako AgriBus.
2. Vyberte vhodný BPM/ESB systém a navrhněte postup jeho nasazení a přizpůsobení tak, aby byl v souladu se specifikací AgriBus, resp. s její částí, která je potřeba pro integraci IS Národní dotace 2016.
3. Po konzultaci s vedoucím navržené řešení implementujte, zprovozněte a demonstrujte možnosti jeho integrace s IS Národní dotace 2016. Ověřte, že zmiňovaný IS funguje se zprovozněným BPM/ESB systém podobně, jako s BPM/ESB AgriBus, a tím prokažte připravenost zmiňovaného IS na budoucí integraci se skutečnou AgriBus.
4. Výsledek zhodnoťte a diskutujte možná vylepšení a rozšíření.

Literatura:

- *Vybudování a provoz komunikační infrastruktury MZe - AgriBus, Zadávací dokumentace příloha č. 4 - Technická specifikace*. Ministerstvo zemědělství ČR, 2014. [https://zakazky.eagri.cz/document_download_17720.html]
- *Veřejná zakázka: Informační systém Národní dotace 2016*, Ministerstvo zemědělství ČR, 2015 [https://zakazky.eagri.cz/contract_display_4591.html]
- Chappell, David A. *Enterprise service bus*. O'Reilly, 2004. ISBN 0-596-00675-6.

Při obhajobě semestrální části projektu je požadováno:

- Body 1, 2 a započatá práce na bodu 3.

Podrobné závazné pokyny pro vypracování diplomové práce naleznete na adrese http://www.fit.vutbr.cz/info/szz/

Technická zpráva diplomové práce musí obsahovat formulaci cíle, charakteristiku současného stavu, teoretická a odborná východiska řešených problémů a specifikaci etap, které byly vyřešeny v rámci dřívějších projektů (30 až 40% celkového rozsahu technické zprávy).

Student odevzdá v jednom výtisku technickou zprávu a v elektronické podobě zdrojový text technické zprávy, úplnou programovou dokumentaci a zdrojové texty programů. Informace v elektronické podobě budou uloženy na standardním nepřepisovatelném paměťovém médiu (CD-R, DVD-R, apod.), které bude vloženo do písemné zprávy tak, aby nemohlo dojít k jeho ztrátě při běžné manipulaci.

Vedoucí: **Rychlý Marek, RNDr., Ph.D.**, UIFS FIT VUT
Datum zadání: 1. listopadu 2015
Datum odevzdání: 25. května 2016

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
Fakulta informačních technologií
Ústav informačních systémů
612 66 Brno, Božetěchova 2

L.S.

doc. Dr. Ing. Dušan Kolář
vedoucí ústavu

Abstrakt

Tato diplomová práce se zabývá návrhem a ukázkou možné integrace Informačního Systému Národní Dotace 2016 (ISND) do plánovaného řešení sběrnice Enterprise Service Bus (ESB) Agribus a procesní platformy Business Proces Management (BPM) Agribus. V této práci je provedena analýza požadavků na ESB a BPM Agribus a výběr vhodného řešení pro jejich implementaci. Pro implementaci ESB AgriBus je vybrán Apache ServiceMix. Dále se tato práce zabývá návrhem změn současného řešení ISND, které je potřeba provést, aby ISND byl připraven na novou roli, kterou bude zastávat v ESB AgriBus. Možnost integrace ISND do ESB Agribus je ukázána na modelovém business procesu životním cyklu žádosti o dotace.

Abstract

This master thesis deals with proposal of design and demonstration of possible integration between Nation Subsidies Information System 2016 (ISND) and future solutions Enterprise Service Bus (ESB) Agribus and process platform Business Process Management (BPM) Agribus. There is performed an analysis of requirements for ESB AgriBus and BPM AgriBus. For the implementation of ESB AgriBus is used Apache ServiceMix. Furthermore, this thesis deals with a proposal of improvements of the current solution of ISND, which are required to perform new role in the ESB AgriBus. The possibility of the integration between ISND and ESB AgriBus is demonstrated on a model business process of application for grants.

Klíčová slova

ESB, BPM , Webové služby, Integrace, Apache ServiceMix, Apache Camel, Activiti BPM, AgriBus, Informační systém Národní dotace

Keywords

ESB, BPM, Web services, Integration, Apache ServiceMix, Apache Camel, Activiti BPM, AgriBus, Nation Subsidies Information System

Citace

ORAVA, Martin. Procesní platforma pro Informační systém Národní dotace. Brno, 2016. Diplomová práce. Vysoké učení technické v Brně, Fakulta informačních technologií. Vedoucí práce Rychlý Marek.

Procesní platforma pro Informační systém Národní dotace

Prohlášení

Prohlašuji, že jsem tuto diplomovou práci vypracoval samostatně pod vedením RNDr. Marka Rychlého, Ph.D. Další informace mi poskytli Ing. Jaroslav Košulič a Ing. Pavel Dvořáček. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

.....
Martin Orava
25.5.2016

Poděkování

Děkuji Ing. Jaroslavu Košuliči a Ing. Pavlu Dvořáčkovi ze společnosti PDS za podporu při tvorbě práce a za užitečné podměty a rady, které mi poskytli.

Dále děkuji RNDr. Marku Rychlému, Ph.D. za to, že se ujal vedení této diplomové práce a za rady které mi poskytl a čas který mi věnoval při konzultacích.

Obsah

1 Úvod

Ministerstvo zemědělství má svou vizi o rozšiřování informačních systémů pod jeho správou. Součástí této vize je ESB AgriBus – sběrnice, která bude sloužit k propojení všech současných i budoucích informačních systémů. Proto již nyní je třeba vyvíjené informační systémy připravit na novou roli, které zavedením sběrnice získají.

V současnosti je vyvíjen Informační systém pro Národní datace 2016 (ISND). Tento systém by měl být připraven pro budoucí zasazení do ESB AgriBus a ke spolupráci s BPM AgriBus.

Cílem této diplomové práce je ukázat, že ISND bude po úpravě s nízkými náklady připraven na novou roli, kterou získá napojením na architekturu ESB AgriBus.

1.1 Postup práce

Abychom dokázali, že ISND může v budoucnosti být rozšířen o novou roli v rámci ESB Agribus, musíme se nejdříve seznámit se zadávací dokumentací na tento systém a zjistit, jaká je jeho plánovaná funkcionalita.

Dále se seznámíme s vizí popisující plánovanou sběrnici ESB AgriBus. Čtenáře seznámíme s hlavním konceptem a požadavky na tuto sběrnici.

Následně porovnáme požadavky na ESB AgriBus s open source ESB řešeními a vybereme vhodnou platformu pro implementaci.

Dále ukážeme, jakým způsobem bude vypadat úprava stávající implementace ISND, aby byl připraven na novou roli, kterou bude plnit v rámci ESB AgriBus.

Vybranou ESB platformu nasadíme a ukážeme možnosti komunikace s ISND na modelovém příkladu.

2 Informační systém národní dotace 2016 (ISND)

Cílem této kapitoly je představení ISND se zaměřením na jeho roli v rámci ESB Agribus. Většina textu v této kapitole je převzata ze zadávací dokumentace tohoto systému [1].

Hlavními funkcemi sytému jsou:

- stanovení zásad a podmínek pro poskytování dotací,
- příjem žádostí o dotace,
- zpracování žádostí o dotace,
- výplata prostředků přiznaných v rámci schválených žádostí o dotace,
- kontrola plnění podmínek stanovených pro jednotlivé dotační programy a podprogramy.

ISND bude volat webové služby publikované ESB integrační komponentou v okamžiku založení žádosti anebo na vyžádání zobrazovat informace v žádosti. Informace o čerpaných podporách za poslední 3 roky bude ISND načítat z registru RDM (Registr podpor malého rozsahu) prostřednictvím rozhraní webových služeb. Integrace s RDM může být dále volitelně řešena využitím integrační komponenty ESB.

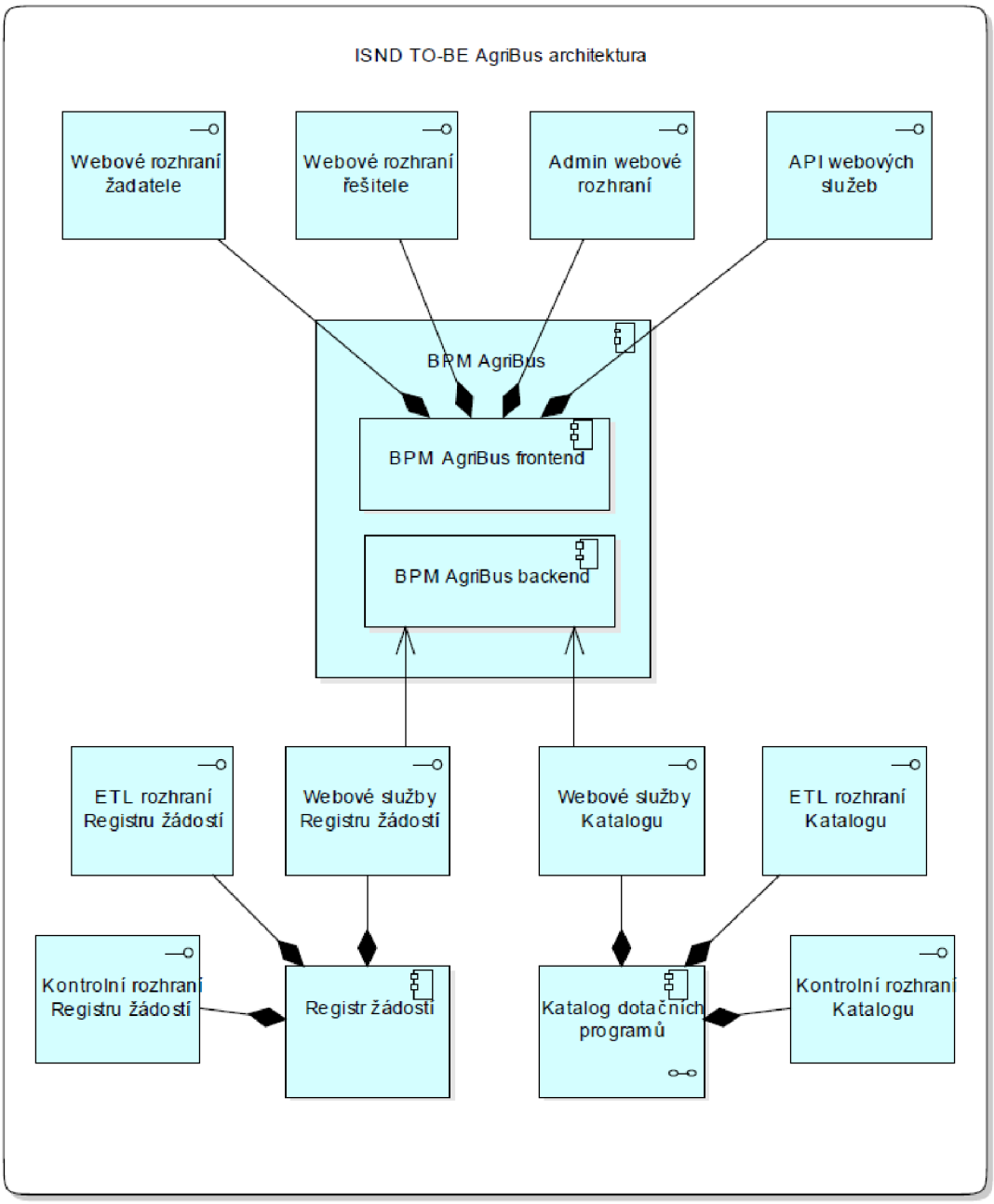
2.1 Podpora dalšího rozvoje ISND

Ministerstvo zemědělství (MZe) připravuje nasazení nové centrální procesní platformy (BPM) a komunikační sběrnice (ESB) AgriBus, která bude centrálním sdíleným systémem pro řízení a provoz procesně orientovaných agend. Po zprovoznění řešení AgriBus by procesní řízení zpracování žádostí o dotace a souvisejících dat mělo být řízeno komponentou AgriBus – BPM AgriBus. Řešení integrované do BPM AgriBus předpokládá rozdělení ISND do třech hlavních modulů:

- workflow BPM platforma řídící procesy a poskytující webové rozhraní pro interakci s uživateli aplikace – BPM AgriBus,
- Katalog dotačních programů přístupný prostřednictvím webových služeb, obsahující pro každý rok přehled dotačních programů a podprogramů a související informace,
- Registr žádostí přístupný prostřednictvím webových služeb, obsahující záznamy o všech podáních v rámci agendy.

Systém ISND by měl v rámci TO-BE architektury zastávat roli **Katalogu dotačních programů** a roli **Registru žádostí**. Obě tyto funkce by měly být přístupné prostřednictvím rozhraní webových služeb.

Vedlejším cílem budovaného řešení ISND je navrhnout takovou architekturu řešení, která umožní s minimálními náklady přechod k TO-BE architektuře zahrnující BPM AgriBus. Navrhovaná architektura ISND by tedy měla umožňovat s minimálními náklady **nahradit prezentační vrstvu webového uživatelského rozhraní prezentační vrstvou webových služeb** volaných BPM AgriBus v TO-BE architektuře.

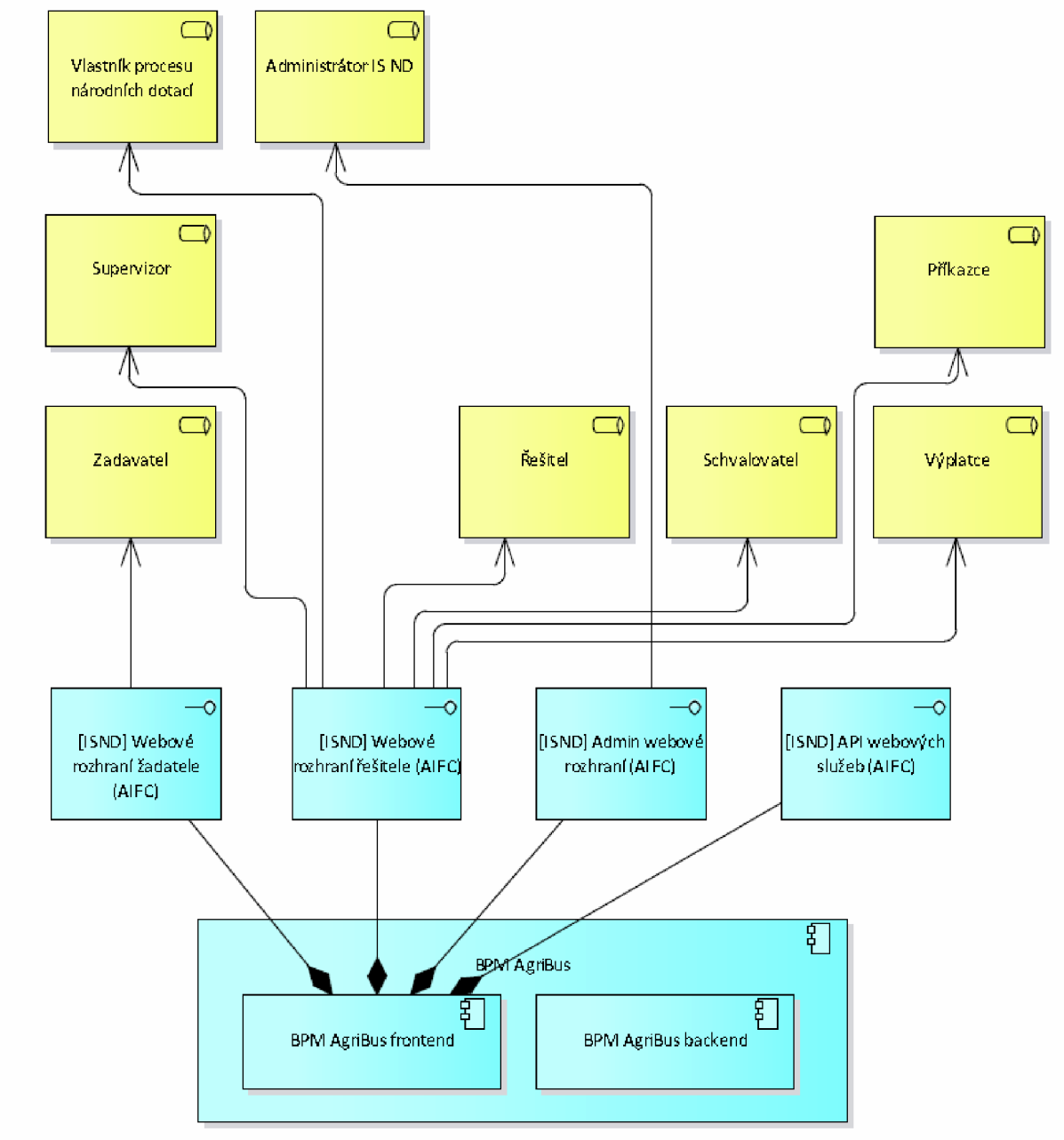


Obrázek 1 - ISND TO-BE AgriBus Architektura

2.2 Přehled TO-BE architektury BPM AgriBus

Tato kapitola obsahuje popis budoucí varianty řešení ISND, která bude realizována po nasazení a zprovoznění BPM AgriBus, respektive popis BPM AgriBus. Specifikace BPM AgriBus uvedená v této kapitole má sloužit jako východisko a zdroj informací pro návrh architektury komponent Registr žádostí a Katalog dotačních programů resp. migraci funkcionalit ISND do funkcionalit těchto komponent.

BPM AgriBus je univerzální procesní platforma (Workflow/BPM systém), který bude zajišťovat řízení procesu zpracování dotačních požadavků, žádostí a souvisejících informací (dále jen procesních případů). Procesní platforma BPM AgriBus bude nabízet všechny standardní funkcionality Workflow/BPM řešení. Pro interakci s uživatelem bude procesní platforma poskytovat několik typů rozhraní ilustrovaných na obrázku - Obrázek 2 - Rozhraní služby BPM AgriBus



Obrázek 2 - Rozhraní služby BPM AgriBus

Pro informaci o existujících dotačních programech a podprogramech, náležitostech a struktuře procesních případů bude procesní platforma využívat služeb aplikační komponenty Katalog dotačních programů přístupných prostřednictvím rozhraní webových služeb. Definice formulářů a informace o datové struktuře procesních případů bude tedy udržována externí komponentou a ukládána/načítána prostřednictvím webových služeb do/z komponenty Katalog dotačních programů.

Obsluhované procesní případy budou ukládány do komponenty Registr žádostí prostřednictvím rozhraní webových služeb.

TO-BE architektura ISND se zapojením BPM AgriBus přepokládá vytvoření dvou hlavních aplikačních komponent:

- Registru žádostí,
- Katalogu dotačních programů.

Obě aplikační komponenty jsou svou povahou databázové aplikace s přístupem prostřednictvím rozhraní webových služeb. Informace o základním databázovém schématu a rozhraní komponent jsou uvedeny dále v dokumentu. Pro komponenty není požadováno vytvoření grafického uživatelské rozhraní. Funkcionalita komponent by v TO-BE architektuře měla být zajištěna novým řešením ISND. Nový systém ISND by tedy měl umožňovat s minimálními náklady nahradit prezentační vrstvu webového uživatelského rozhraní prezentační vrstvou webových služeb a umožnit přechod k TO-BE architektuře.

Procesní případy budou řízeny dle předem definovaných workflow, přičemž každému typu procesního případu bude přiřazeno právě jedno workflow. Workflow bude tvořeno stavy procesního případu a přechody mezi stavy.

Pro stavy bude možné definovat zejména:

- povinná pole procesního případu v rámci stavu,
- jaké výchozí vstupní formuláře mají být ve stavu použity,
- výchozí odpovědný útvar a výchozí odpovědná osoba za procesní případ ve stavu,
- lhůty pro obsluhu případů v daném stavu,
- eskalační pravidla v případě prodlení či splnění jiných definovaných podmínek.

Pro přechody mezi stavy bude možné definovat zejména:

- jaké musí být splněny podmínky pro přechod do následného stavu,
- jaké akce se mají před přechodem nebo po přechodu do stavu vykonávat.

Akce, které bude možné spustit při přechodu mezi stavy, budou zahrnovat:

- odeslání notifikace e-mailem,
- změna procesního případu,
- předání informací do externího systému využitím integračního rozhraní (uvažovaná integrační rozhraní jsou popsána dále v dokumentu),
- zavolání webové služby s možností předání informací z procesního případu jako parametrů.

Pro e-mailové notifikace bude možné definovat šablony notifikace, které budou moci obsahovat zejména:

- informace z podkladového záznamu (hodnotu z jakéhokoli pole záznamu),
- formátovaný text,

- přílohy z podkladového záznamu,
- hypertextový odkaz na záznam v Procesní platformě.

Procesní platforma bude řídit proces obsluhy různých typů požadavků a žádostí s rozdílnou strukturou informací (procesních případů). Systém bude podporovat návrh a modifikaci definice datové struktury procesních případů a návrh formulářů pro jednotlivé typy procesních případů. Systém dále bude podporovat uložení definice struktury datové věty procesních případů, včetně doplňujících informací a formulářů do externí komponenty Katalog dotačních programů prostřednictvím webových služeb v okamžiku založení nového procesního případu. Dále umožní vyčíst informace o struktuře dat procesního případu a formulářích z komponenty Katalog dotačních případů. Definice typů procesních případů a všech souvisejících konfigurací bude realizována proškoleným administrátorem na straně zadavatele a bude tedy realizována cestou konfigurace/přizpůsobení systému bez potřeby zásahu do zdrojového kódu platformy.

2.2.1 Uvažovaná dostupnost, výkonnost a kapacity

Předpokládané využití agendy ISND v rámci Procesní platformy BPM AgriBus je následující:

- maximálně 50 000 nových anebo měněných procesních případů ročně,
- maximální zatížení systému ve špičce činí 300 nových anebo měněných procesních případů za hodinu,
- předpokládaný maximální roční objem dat činí 300 GB.

Parametry dostupnosti agendy v rámci BPM AgriBus jsou následující:

- provozní doba služby 7x24,
- dostupnost služby 98 %.

Dostupnost a provozní doba samotné platformy BPM AgriBus budou 7x24 a 99,9 %.

2.2.2 Přehled základních funkcionalit poskytovaných BPM AgriBus

Procesní platforma bude poskytovat zejména následující funkcionality:

- webové uživatelské rozhraní,
- webové anebo desktopové administrátorské rozhraní zahrnující funkcionality pro definici workflow/procesů a formulářů,
- podporu workflow pro řízení procesních případů zahrnujících stavy procesních případů, přechody mezi stavy a další výše uvedené informace,

- podpora různé struktury dat jednotlivých typů procesních případů a ukládání a načítání struktury a souvisejících formulářů do/z externí aplikační komponenty prostřednictvím webových služeb,
- podpora ukládání a načítání dat procesních případů do/z externí aplikační komponenty prostřednictvím webových služeb,
- podpora definice sdílených číselníků,
- možnost definice časových lhůt pro jednotlivé stavy a typy procesních případů, hlídání lhůt a eskalace v případě prodlení,
- možnost připojovat k procesním případům komentáře s určením autora, data a času pořízení komentáře,
- podpora notifikací a notifikačních šablon v rámci workflow,
- možnost vykonání automatických akcí v rámci workflow výše uvedených typů,
- možnost přidávat k procesním případům souborové přílohy a odkazovat přílohy v externích umístěních,
- podpora schvalování procesních případů změnou stavu případu, nastavením atributu případu a cestou schvalovacího workflow (např. využitím schvalovacích tiketů), podpora paralelního a sériového schvalování na více úrovních,
- podpora auditování změn záznamů a logování vykonaných eskalací a automatických akcí,
- podpora šablon reportů, sestav, exportů (PDF, XLSX, CSV) a tisku.

2.2.3 Přehled funkcionalit rozhraní BPM AgriBus

Všechna níže uvedená webová uživatelská rozhraní poskytují následující základní funkcionality:

Pro přihlášení a ověřování uživatelů:

- přihlášení a odhlášení uživatele,
- přístup prostřednictvím SSL,
- lokalizace do českého jazyka,
- správa uživatelského profilu a změna hesla,
- webové rozhraní bude integrováno do portálového řešení MZe. Autentizaci přistupujícího uživatele provede portálové řešení a jeho identitu předá webovému rozhraní žadatele prostřednictvím SSO hlavičky. Pro žadatele budou v Procesní platformě vydefinovány role omezující práva a tyto role zrcadleny v LDAP MZe. Procesní platforma bude umožňovat import uživatelů a rolí z AD/LDAP,
- možnost nastavení notifikací, resp. kdy, jakou formou a kam mají být notifikace zasílány.

Pro editaci procesních případů:

- přehledné webové formuláře pro přidávání a editaci procesních případů podporující rozdělení informací do sekcí a záložek,
- načtení definice formulářů z externí komponenty prostřednictvím webových služeb,
- automatický výběr vhodného formuláře dle kontextu: přihlášený uživatel, informace v procesním případu, stav procesního případu,
- podpora dynamického překreslování formulářů v závislosti na průběžně vybraných hodnotách v polích formulářů, podpora běhu skriptu na pozadí formulářů a modifikace formulářů a hodnot běžícím skriptem,
- zvýraznění povinných polí v příslušném stavu workflow,
- provedení kontroly vstupních dat před uložením a zvýraznění chyb ve formuláři,
- možnost automatického doplnění polí ve formuláři z číselníků (typicky na základě hodnot ostatních polí ve formuláři),
- tabulkový pohled na přehled procesních případů,
- možnost definice zobrazených polí, filtrů, řazení a seskupování procesních případů s možností uložení pro opakované použití, podpora soukromých, veřejných a povinných pohledů na data,
- možnost grafického zvýraznění případů na základě hodnot polí procesního případu,
- možnost definice front procesních případů,
- dashboard pro prezentaci počtů procesních případů v jednotlivých frontách,
- možnost přidání procesních případů do oblíbených položek a správa oblíbených položek,
- vyhledávání procesních případů prostřednictvím vyhledávacího formuláře se zástupnými znaky a prostřednictvím full-text hledání.

Pro sestavy a tisk:

- možnost definice vlastních přehledů a vygenerování reportů z předdefinovaných šablon,
- podpora souhrnů a výpočtu hodnot v rámci přehledů,
- podpora exportu do PDF, XLSX a CSV,
- možnost definice zobrazených polí, filtrů, řazení a seskupování procesních případů s možností uložení zobrazení pro opakované použití, podpora soukromých, veřejných a povinných zobrazení dat.
- možnost automatického doplnění polí ve formuláři na základě provedeného volání externí webové služby, přičemž metodě webové služby mohou být jako parametry předány hodnoty polí z procesního případu, doplnění může být spuštěno manuálně uživatelem z kontrolního prvku (typicky tlačítko) na formuláři anebo automatickou akcí v rámci procesu.

Pro prezentaci procesních případů:

- tabulkový pohled na přehled procesních případů,
- možnost definice zobrazených polí, filtrů, řazení a seskupování procesních případů s možností uložení pro opakované použití, podpora soukromých, veřejných a povinných pohledů na data,
- možnost grafického zvýraznění případů na základě hodnot polí procesního případu,
- možnost definice front procesních případů,
- dashboard pro prezentaci počtů procesních případů v jednotlivých frontách,
- možnost přidání procesních případů do oblíbených položek a správa oblíbených položek,
- vyhledávání procesních případů prostřednictvím vyhledávacího formuláře se zástupnými znaky a prostřednictvím full-text hledání.

2.2.4 Webové rozhraní žadatele

Webové rozhraní žadatele je určeno pro koncové žadatele. V roli žadatele mohou být zaměstnanci MZe, kooperující organizace či samotní žadatelé, resp. uživatelé, kteří nejsou zaměstnanci zadavatele ani kooperujících organizací. Obsah rozhraní bude omezen na níže poskytované funkcionality.

Požadavky na funkcionality

Webové rozhraní bude obsahovat veřejnou část, pro kterou nebude požadováno přihlášení do aplikace, a neveřejnou část, k níž bude přistupováno přihlášením se využitím uživatelského jména a hesla.

Veřejná část bude poskytovat zejména:

- nahlížení a vyhledávání v Katalogu dotačních programů,
- možnost stažení tiskových formulářů a tisku tiskových formulářů,
- nástěnku s veřejnými informacemi pro všechny potenciální žadatele.

Neveřejná část rozhraní bude poskytovat zejména:

- editace profilu a nastavení notifikace,
- přístup k založeným požadavkům a žádostem (procesním případům),
- přístup k vygenerovaným notifikacím,
- možnost založení zcela nového požadavku anebo požadavku z existujícího procesního případu s kopií hodnot,
- možnost elektronicky autorizovat požadavek (informace o autorizaci požadavků jsou uvedeny v části Přehled uvažovaných integrací),
- možnost vytisknout vyplněný požadavek včetně autorizačního hash.

2.2.5 Webové rozhraní řešitele

Webové rozhraní řešitele je určeno pro zaměstnance MZe anebo kooperujících organizací vystupujících v procesech obsluhy žádostí.

Funkcionality rozhraní

Mimo obecných požadavků na webové rozhraní musí rozhraní nabídnout následující funkcionality přístupné pouze po řádném přihlášení do aplikace:

- dostupné části aplikace a akce pro uživatele v závislosti na jeho roli v procesech (příslušnosti k uživatelským skupinám),
- podpora modifikace procesní části dat procesních případů,
- možnost komentování procesních případů,
- podpora schvalování procesních případů,
- podpora spuštění manuálních akcí nad procesními případy.

2.2.6 Admin webové nebo desktop rozhraní

Admin webové rozhraní bude sloužit pro administraci komponenty Procesní platformy.

Funkcionality rozhraní

Admin webové rozhraní bude poskytovat zejména následující hlavní funkcionality:

- základní konfigurace platformy včetně správy uživatelských účtů, rolí a oprávnění,
- grafická konfigurace procesů a workflow (změny, přidávání nových workflow) a všech jeho výše uvedených složek,
- konfiguraci přidělení workflow jednotlivým typům procesních případů evidovaným v externí komponentě Katalog dotačních programů,
- konfigurace eskalačních lhůt pro individuální typy případů pro jednotlivé stavy v rámci workflow,
- grafický návrh formulářů využívající informace o struktuře dat z Katalogu dotačních programů,
- grafický návrh tiskových sestav umožňující plnou kontrolu nad vzhledem a rozmístěním komponent tiskové sestavy,
- uložení definice formulářů do externí komponenty prostřednictvím rozhraní webových sužeb,
- možnost využít v rámci definice formulářů opakovaně použitelné sub-formuláře,
- možnost v rámci definice datové struktury procesního případu využít opakovaně použitelnou sub-strukturu,
- podpora obvyklých datových typů v procesních případech a podpora odpovídajících vstupních prvků ve formulářích (text, combo box. list, atd.),

- podpora referenčních polí procesního případu resp. odkazů mezi různými procesními případy a odkazů na číselníky typů: 1:1, 1:N a M:N,
- definice auditovaných polí procesních případů,
- definice polí zahrnutých do full-text vyhledávání.

2.2.7 API webových služeb

API webových služeb bude poskytovat rozhraní pro práci s procesními případy z externích aplikací.

Funkcionality rozhraní

Základní funkcionality API rozhraní jsou:

- autentifikace a autorizace uživatelů oproti LDAP/AD,
- zabezpečený přístup SSL,
- podpora standardů SOAP, UDDI, WSIL a WSDL,
- podpora CRUD operací nad procesními případy,
- logování operací stejným způsobem jako v případě skutečného uživatele systému.

2.3 Přehled uvažovaných integrací

V rámci řešení jsou uvažovány následující integrace:

- e-mailový SMTP server zadavatele,
- odesílání e-mailových notifikací,
- LDAP/AD zadavatele,
- autentifikace a autorizace uživatelů, import uživatelů a skupin,
- provozní a bezpečnostní monitoring systémy zadavatele,
- datové schránky.

Procesní platforma v okamžiku autorizace požadavku vygeneruje kontrolní hash z informací v procesním případu, hash uloží do datové věty procesního případu a kopii hash nabídne žadateli pro odeslání datovou schránkou. Pro odeslání může být použito API webových služeb Informačního systému Datových schránek. XML zpráva odesílaná datovou schránkou bude obsahovat mimo dalších informací jednoznačný identifikátor procesního případu, ke kterému se autorizace váže, a identifikaci, že se jedná o autorizační zprávu.

Procesní platforma bude pravidelně vyčítat prostřednictvím integračních webových služeb zadavatele datovou schránku a stahovat autorizační zprávy. Platforma následně ověří shodu hash v procesním případu. V případě shody nastaví příznak autorizace procesního případu na autorizovaný a zajistí propagaci informací z požadavku do žádosti.

- Rozhraní IS státní pokladny

  Manuální export na základě sestavených přehledů a souhrnů.
- IS EDS/SMVS šifrované dávky

  Manuální export na základě sestavených přehledů a souhrnů.
- CEDR rozhraní

  Transportním protokolem pro popisované rozhraní bude protokol SMTP-MIME a XML protokol SOAP ve verzi 1.2. Export bude prováděn manuálně pravděpodobně jednou ročně.
- Registr podpor de minimis

  Integrace bude realizována prostřednictvím volání webových služeb integrační platformy zadavatele. Volání bude realizováno jako automatická akce v rámci workflow.
- ETL rozhraní datového skladu

  Manuální export jednou ročně na základě sestavených přehledů a souhrnů.

2.4 Přehled TO-BE architektury AgriBus

V této kapitole popíšeme budoucí varianty řešení ISND, která bude realizována po nasazení a zprovoznění BPM AgriBus, respektive specifikaci komponent Registr žádostí a Katalog dotačních programů. Systém ISND by měl po nasazení BPM AgriBus převzít roli komponent Registr žádostí a Katalog dotačních programů. Specifikace uvedená v této kapitole a podkapitolách bude použita pro definici alternativní/budoucí TO-BE architektury ISND integrované do BPM AgriBus.

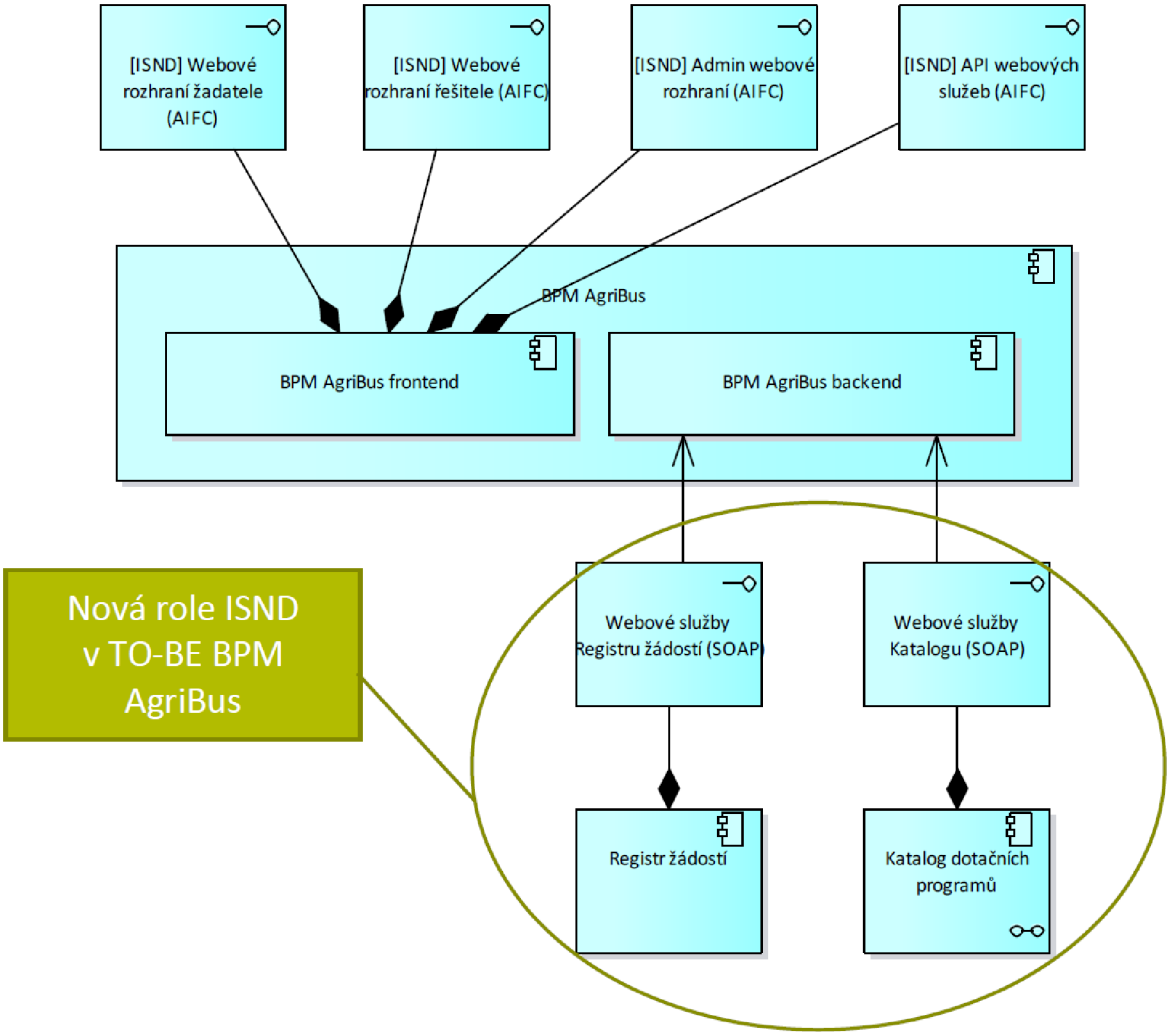
TO-BE architektura ISND se zapojením BPM AgriBus přepokládá vytvoření dvou hlavních aplikačních komponent:

- Registru žádostí,
- Katalogu dotačních programů.

Obě aplikační komponenty jsou svou povahou databázové aplikace s přístupem prostřednictvím rozhraní webových služeb. Informace o základním databázovém schématu a rozhraní komponent jsou uvedeny dále v dokumentu. Pro komponenty není požadováno vytvoření grafického uživatelské rozhraní. Funkcionalita komponent by v TO-BE architektuře měla být zajištěna novým řešením ISND. Nový systém ISND by tedy měl umožňovat s minimálními náklady nahradit prezentační vrstvu webového uživatelského rozhraní prezentační vrstvou webových služeb a umožnit přechod k TO-BE architektuře.

Webové služby poskytované aplikačními komponentami budou využívány ostatními aplikačními komponentami ISND AgriBus a dalšími komponentami tvořícími aplikační infrastrukturu zadavatele anebo kooperujících organizací viz Obrázek 3 - Nová role ISND v TO-BE BPM AgriBus.

Přístup k webovým službám z aplikačních komponent mimo ISND AgriBus bude typicky zprostředkován centrální komunikační sběrnicí (ESB) zadavatele.



Obrázek 3 - Nová role ISND v TO-BE BPM AgriBus

2.4.1 Databáze dotačních žádostí

Databáze dotačních žádostí je aplikační komponenta odpovědná za skladování požadavků a žádostí a všech souvisejících informací. Databáze bude publikována prostřednictvím webových služeb pro ostatní aplikační komponenty tvořící aplikační infrastrukturu zadavatele nebo pro komponenty kooperujících organizací. Databáze bude obsahovat veškeré informace nezbytné pro řádné obsloužení žádostí. Informace budou děleny do dvou základních skupin:

- **náležitosti žádosti** - zahrnují veškeré informace, za jejichž správnost a úplnost odpovídá žadatel. Tyto informace mohou být v systému pořízeny, aktualizovány či mazány výhradně prostřednictvím požadavků autorizovaných žadatelem.
- **procesní informace** - zahrnují informace pořízené v průběhu obsluhy a zpracování žádosti a slouží primárně jako podpůrné informace samotného procesu zpracování žádostí. Tyto

informace mohou být měněny osobou v roli oprávněné k modifikaci v rámci procesu bez autorizace žadatelem.

Registr žádostí bude obsahovat jak samotné žádosti o dotace a související informace, tak dílčí požadavky na založení, modifikaci, doplnění či zrušení žádostí. Autorizovaný požadavek bude v databázi opatřen kontrolním hash. Pro plnou autorizaci požadavku musí být kontrolní hash ověřen. Generování a ověřování hash bude probíhat mimo Databázi žádostí. Autorizovaný požadavek bude propagován do změny datové věty žádosti.

Struktura informací vázajících se k jedné žádosti v databázi závisí na typu požadavku a žádosti resp. na dotačních programů a podprogramů, v rámci něhož byla žádost podána. Typy požadavků, žádostí, dotační programy a podprogramy a s nimi související struktura se v čase (typicky meziročně) mění. Jedním z hlavních cílů je navrhnout takový systém, který bude poskytovat univerzální konfigurovatelné funkcionality umožňující efektivně měnit a vytvářet struktury pro evidenci žádostí bez nutnosti zásahu do implementace či programového kódu systému.

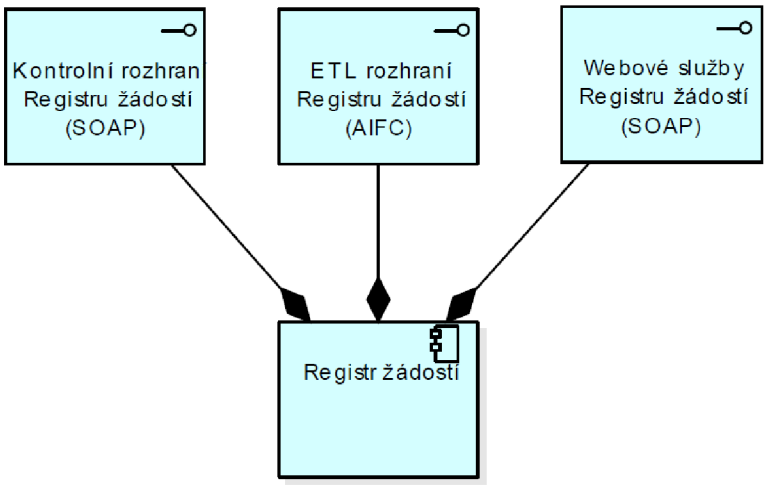
Přehled základních funkčních požadavků

Databáze dotačních žádostí musí poskytovat tyto hlavní funkcionality:

- zajištění evidence všech požadavků a žádostí a souvisejících entit s různou strukturou informací,
- přístup k informacím v databázi prostřednictvím dále uvedených rozhraní,
- možnost konfigurace struktury požadavků, žádostí či přidání nových typů žádostí či požadavků prostřednictvím webových služeb,
- podpora uložení autorizačních hash k požadavkům zahrnujících informace z požadavku a datum a čas přijetí požadavku,
- evidence stavu autorizace požadavků,
- propagace změn navrhovaných v rámci požadavku do žádosti,
- podpora evidence příloh k požadavkům a žádostem ve formě přiložených souborů anebo odkazů na soubory v externích umístěních,
- zajištění konkurenčního přístupu k editovaným žádostem a požadavkům.

Přehled rozhraní a požadavků na rozhraní

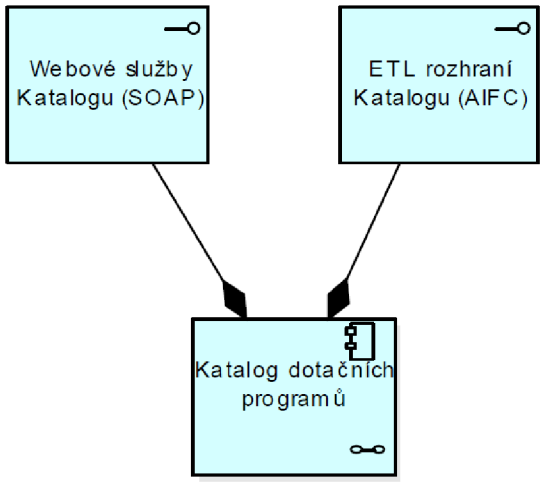
Služby Registru žádostí by měly být poskytovány prostřednictvím rozhraní ilustrovaných na obrázku - Obrázek 4 - Rozhraní Registru žádostí



Obrázek 4 - Rozhraní Registru žádostí

2.4.2 Katalog dotačních programů

Katalog dotačních programů je aplikační komponenta zajišťující evidenci dotačních programů, podprogramů a souvisejících informací. Komponenta poskytuje služby prostřednictvím rozhraní uvedených na obrázku - Obrázek 5 - Rozhraní Katalogu dotačních programů. Hlavním rozhraním pro přístup k funkcionalitám komponenty je rozhraní webových služeb. Přehled rozhraní komponenty a popis požadovaných funkcionalit rozhraní je uveden dále v dokumentu.



Obrázek 5 - Rozhraní Katalogu dotačních programů

Struktura informací vázajících se k jednomu dotačnímu podprogramu závisí na typu podprogramu. Každý podprogram může obsahovat řadu dotačních položek typu sazbová položka anebo podílová položka včetně kombinace obou. Podílová položka může být dále omezena sazbou. Cílem řešení je navrhnout takový datový model a funkcionality komponenty Katalog dotačních programů, které budou umožňovat meziročně přidávat a spravovat nové typy dotačních programů, podprogramů a položek prostřednictvím rozhraní webových služeb a ETL rozhraní (popis dále v dokumentu) bez programátorských zásahů do implementace či programového kódu systému.

Pro upřesnění představy o struktuře jednotlivých programů a podprogramů lze využít dokument Zásady pro poskytování dotací pro rok 2015 dostupné na adrese http://eagri.cz/public/web/file/361919/Zasady_pro_rok_2015.doc.

Komponenta Katalog dotačních programů bude dále nabízet služby výpočtu dotačních nároků dostupné prostřednictvím webového rozhraní. Výpočet dotačního nároku bude využívat vzorec pro výpočet nároku evidovaný v rámci dotačního podprogramu. Vzorec bude obsahovat matematickou formuli (případně komplexnější skript) pro výpočet nároku dotace z informací uvedených v připojených záznamech typu Požadovaná informace a z informací uvedených v sazbových položkách. Každá požadovaná informace a sazbová položka bude opatřena jednoznačným identifikátorem, který bude možné odkazovat ve formuli. Komponenta by při založení nového dotačního podprogramu nebo modifikaci podprogramu měla ověřit syntaktickou správnost a použitelnost matematické formule.

Přehled základních funkčních požadavků

Katalog dotačních programů musí poskytovat tyto hlavní funkcionality:

- zajištění evidence všech dotačních programů, podprogramů a souvisejících informací s různou strukturou datové věty,
- poskytování služeb pro zadání nových, změnu či zrušení dotačních programů, podprogramů,
- poskytování informací o dotačních programech, podprogramech a souvisejících entitách,
- poskytování příloh (zejména tiskových formulářů) programů a podprogramů,
- vyhledávání v dotačních programech a podprogramech,
- kopírování dotačních programů a podprogramů s možností výběru kopírovaných atributů,
- podpora evidence příloh k programům a podprogramům,
- výpočet dotačních nároků,
- zajištění konkurenčního přístupu k editovaným informacím.

Přehled rozhraní a požadavků na rozhraní

Služby Katalogu dotačních programů by měly být poskytovány prostřednictvím rozhraní ilustrovaných na obrázku Obrázek 4 - Rozhraní Registru žádostí.

3 AgriBus

V této kapitole představíme vizi MZe pro integraci současně používaných informačních systému. Většina textů této kapitoly jsou vhodné výňatky ze zadávací dokumentace na tuto sběrnici. [2]

Základ IT infrastruktury založené na SOA bude i nadále tvořit informační sběrnice služeb ESB AgriBus (Enterprise Service Bus), která spojuje a zprostředkovává všechny komunikace a interakce mezi Poskytovateli a Konzumenty služeb, viz Obrázek 6 - Komunikační platforma ESB. ESB AgriBus platforma bude umožňovat rychle měnit služby, snadno je připojovat, publikovat a řídit. Řešení AgriBus bude dále poskytovat funkcionality obecné procesní platformy (BPM, dále jen AgriBus BPM) umožňující navrhovat a řídit procesy zahrnující lidské vstupy a interakce. BPM AgriBus se stane jednotnou standardizovanou platformou pro návrh a provoz agendových informačních systémů a editorských systémů registrů MZe. BPM AgriBus bude pro komunikaci s ostatními systému využívat ESB AgriBus. Dodávku jednotlivých procesů či agendových systémů implementovaných v BPM AgriBus mohou zajišťovat různí dodavatelé. Zapojení individuálních dodavatelů neohrozí stabilitu a výkonnost BPM AgriBus ani ESB AgriBus.

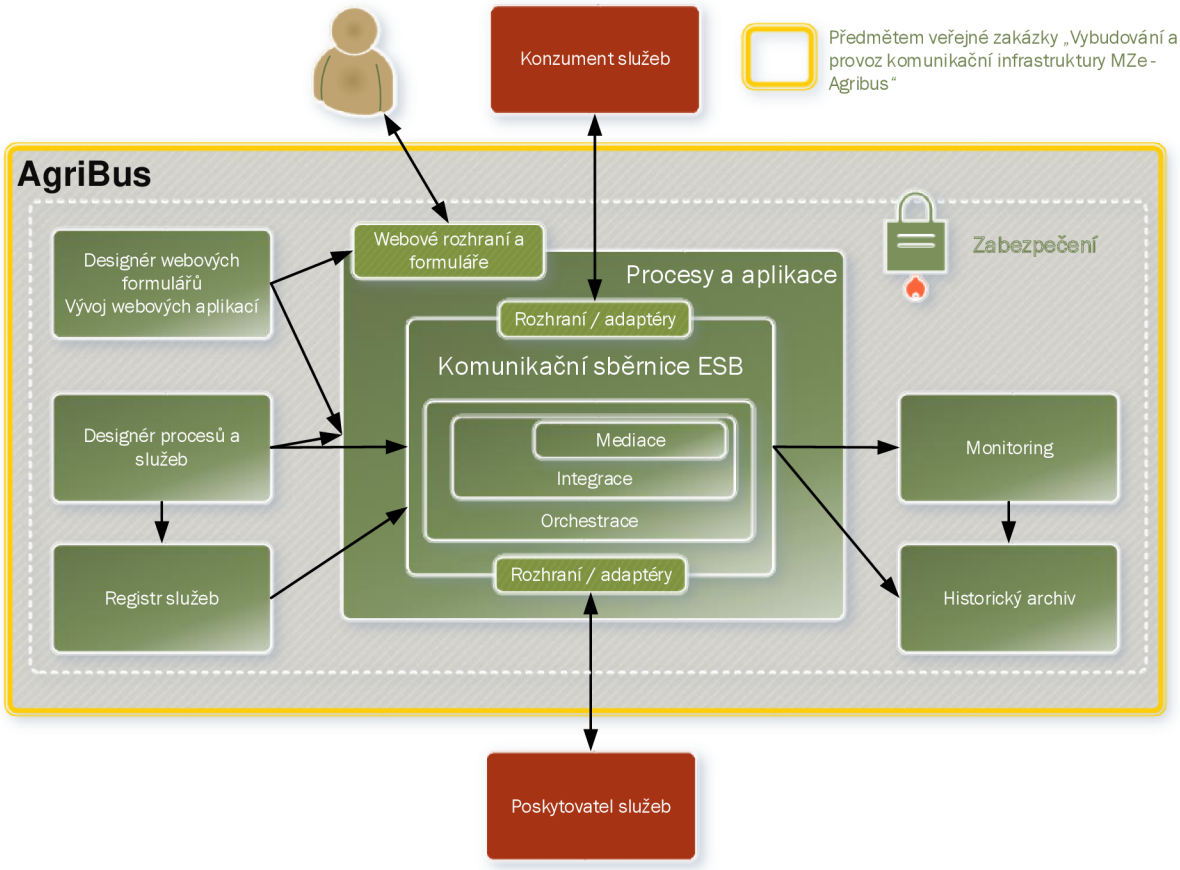
Obrázek 6 - Komunikační platforma ESB

Obrázek 7 - BPM AgriBus

3.1 Popis požadovaného řešení

Informace v této kapitole popisují požadované cílové řešení, které bude aplikováno pro nově implementované služby a nezahrnuje tedy architekturu nezbytnou pro migraci stávající implementace služeb do nového řešení AgriBus (pro paralelní souběh AgriBus s ESB Oracle, pokud je taková architektura trvale či dočasně vyžadována). Níže uvedená architektura bude doplněna tak, aby obsahovala veškeré prvky nezbytné pro migraci všech služeb provozovaných na stávající platformě ESB Oracle.

Řešení AgriBus představuje jednotnou komunikační platformu složenou z několika funkčních komponent. Logický model řešení obsahuje grafické znázornění funkčních komponent řešení a jejich vzájemné spolupráce. Jednotlivé funkční komponenty mohou být v rámci řešení implementovány jedním či více nabízenými systémy. Níže uvedený souhrn řešení popisuje jednotlivé funkční komponenty tak, jak jsou uvedeny na obrázku Obrázek 8 - Logický model AgriBus.



Obrázek 8 - Logický model AgriBus

3.2 Komunikační sběrnice ESB AgriBus

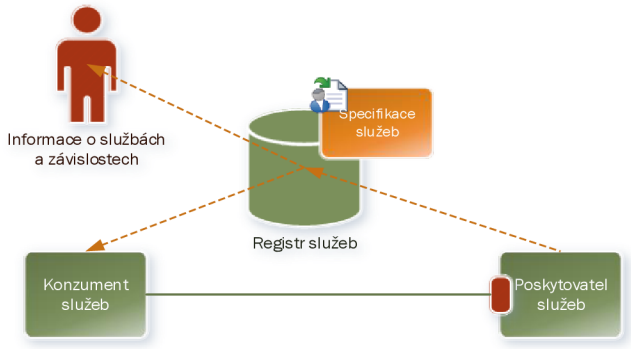
Komunikační vrstva ESB AgriBus je centrální komponentou zajišťující přenos zpráv mezi Poskytovateli a Konzumenty služeb. ESB AgriBus zprostředkovává komunikaci a provádí konverzi mezi formáty a protokoly vstupních požadavků vznesených příjemci služby a formáty a protokoly požadovanými Poskytovateli služby. Pro komunikaci mezi jednotlivými příjemci a Poskytovateli služby není třeba individuálně dojednávat komunikační rozhraní. ESB AgriBus zajišťuje transparentnost komunikace poskytnutím univerzálního komunikačního rozhraní. Příjemci služeb nemusí znát adresu komunikačního rozhraní Poskytovatelů služeb. Adresovatelnost požadované služby a směrování zpráv zajišťuje ESB AgriBus.

3.3 BPM AgriBus

AgriBus bude poskytovat funkcionality a nástroje pro řízení komplexních procesů a pro budování procesně orientovaných agendových nebo editorských aplikací. Součástí řešení bude prostředí zajišťující běh dlouhotrvajících procesů zahrnujících uživatelské interakce. Za tímto účelem bude AgriBus obsahovat funkcionalitu pro návrh, vytváření a prezentaci webových formulářů a aplikací zprostředkujících uživatelské vstupy v procesu. Procesy a formuláře bude možné navrhovat v grafickém prostředí a zároveň přímou editací definičních souborů. AgriBus bude připraven na postupné přidávání/budování výše zmíněných procesů a procesních aplikací (agendových systémů). Běh procesů a aplikací nebude mít negativní dopady, resp. nesmí ovlivnit nebo ohrozit provoz a výkonnost komunikační sběrnice AgriBus. Uživatelské webové rozhraní procesních aplikací bude integrováno do portálového řešení Objednatele. Řešení bude dále podporovat přístup prostřednictvím mobilních platforem (zejména Android, IOS).

3.4 Registr služeb

Registr služeb je komponentou řešení AgriBus poskytující informace o dostupných komponentách SOA architektury. Registr obsahuje katalog služeb a komponent, jejich specifikaci a metadata. Konzument využívá registr služeb k vyhledání jemu dostupných služeb a komponent a k získání specifikace nezbytné pro pochopení jakým způsobem a s jakými parametry je třeba službu volat. Registr obsahuje informace o funkcionalitách poskytovaných jednotlivými službami. Vývojáři nových služeb před návrhem nových služeb mohou jednoduše prohledat registr, zda již neexistuje služba vyhovující jejich požadavkům, čímž je zabráněno nechtěné duplicitě služeb.



Obrázek 9 - Registr služeb

Registr služeb dále podporuje řízení životního cyklu všech evidovaných komponent. O každé evidované komponentě je v rámci metadat možné uchovávat informace, v jakém stavu z pohledu nasazení se komponenta nachází (ve vývoji, v testu, v produkci atd.), verze jednotlivých komponent a dynamický odkaz na poslední zveřejněnou verzi komponenty určenou pro využití v rámci implementace služeb. Registr služeb je přístupný nejen samotným servisním komponentám prostřednictvím aplikačního rozhraní, ale i vybraným uživatelům využitím obsaženého webového rozhraní. Oprávnění uživatelé tak mají možnost získat další informace o komponentách, jako jsou přehledy změn komponent, doporučení pro využití komponent, závislosti komponent, informace o využití komponent atd. Katalog služeb je obsahově provázán s EAR (EAR, není předmětem projektu AgriBus) , tzn. ke každé službě v registru služeb bude existovat dokumentace architektury a modely v EAR. Katalog služeb společně s informacemi z EAR a CMDB bude možné využít například při plánování změn či odstávek vybraných komponent a určení potenciálního dopadu na nadřazené komponenty či služby, které měněné či odstavované komponenty využívají

Služby budou v rámci přípravy Migračního plánu rozděleny do kategorií dle složitosti a komplexnosti. Informace o kategorii služby bude evidovaná v registru služeb ve formě metadat a přístupná v rámci reportingu za účelem výpočtu a vyhodnocení plnění SLA dle katalogového listu služby (pro monitoring dostupnosti a výpočet SLA budou použity služby S1). Rozdělení služeb provede Zhotovitel, přičemž rozdělení podléhá revizi a schválení Objednatelem.

3.4.1 ESB AgriBus

Mezi základní požadavky kladené na komunikační vrstvu ESB AgriBus patří:

Konzistentní práce se službami

- Integrace systémů na bázi webových služeb, umožňujících implementaci SOA.
- Integrace koncových bodů využívajících různé komunikační protokoly (např. jeden koncový bod vyžaduje JMS a druhý koncový bod podporuje pouze webové služby).
- Podpora transportů HTTP, JMS, JCA, Enterprise Java Bean, web services (SOAP 1.2, SOAP 1.1, SOAP 1.1 using JAX-RPC, SOAP 1.1/JMS).

- Podpora orchestrace služeb (řízení workflow služeb modelované v BPEL).
- Integrace s centrálním registrem služeb, jednotný způsob zveřejnění služby, jehož součástí je i popis služby nezávislý na technologii.
- Dynamické vyhledávání služeb v centrálním registru.
- Standardizovaný způsob volání služby (jak synchronními, tak asynchronními protokoly) nezávislý na transportním médiu a technologii služby.
- Integrace s centrálním registrem služeb, jednotný způsob zveřejnění služby, jehož součástí je i popis služby nezávislý na technologii.

Směrování zpráv

- Virtualizace koncových bodů. Konzument služby nemusí znát přesnou aktuální adresu koncového bodu Poskytovatele služby. Konzument používá relativní název Poskytovatele služby a ESB AgriBus rozhoduje, jaký koncový bod a adresu použít (ESB AgriBus k tomuto účelu využívá nastavená kritéria a registr služeb).
- Adresace služeb nezávislá na síťové topologii.
- Dynamické směrování podle obsahu zpráv či podle QoS kritérií (například zatížení) a politik.
- Transparentní přepínání cílových bodů za běhu, výběr cílového bodu z registru služeb dle výše uvedených principů.
- Možnost opakovaného testování využití koncového bodu a volba jiného vhodného bodu z registru služeb v případě selhání volání primárního koncového bodu.
- Podpora standardizovaných adaptérů a konektorů v podobě SCA komponent (např. e-mail, sftp, JDBC).
- Logika pro připojení k cílovému prostředí je zapouzdřena v komponentě adaptéru.
- Standardizovaná rozhraní pro přístup ke sběrnici z různých operačních systémů a technologií.

Mediace a transformace

- Kanonický formát zpráv, kanonický formát datových typů, podpora a správa verzí kanonických formátů.
- Podpora mediace a transformace zpráv pro případ, kdy komunikující body reprezentují shodná data rozdílnými datovými strukturami.
- Obohacování zpráv o dodatečné informace.
- Podpora mediace zpráv na všech komunikačních úrovních.
- Messaging operace.
- Podpora synchronního i asynchronního volání služeb.
- Frontování požadavků, práce s frontami.
- Možnost selektivně zastavit příjem požadavků pro vybrané fronty.

- Zprostředkování komunikace využívající standardní formáty, jako jsou EDI a odstínění Konzumentů od detailů těchto standardů (zapouzdření datových standardů).

Podpora přenosu souborů

- Možnost přenosu objemných souborů v řádu 100 MB a více mezi různými systémy a platformami bez dopadu na kvalitu ostatních komunikačních toků (mimo tělo zprávy).
- Využití datové úschovny pro přenos souborů.
- Dynamické určení cesty pro uložení souborů.
- Přenos souborů do určené cesty a předání adresy souboru Konzumentovi v rámci zprávy s tím, že adresa souboru musí být nepredikovatelná (např. obsahují token random 128 bit base64).
- Monitoring přenosu souborů.
- Volitelné zajištění end-to-end integrity souborů s možností přidat do řídící zprávy hash anebo podpis souboru a podpora přenosu souborů šifrovaných nebo podepsaných Poskytovateli a Konzumenty.
- Volitelné potvrzení přenesení souborů zprávou.
- Oprávnění a zabezpečení na úrovni uživatelských rolí pro přístup k přenášeným souborům.
- Volitelné end-to-end šifrování přenášených souborů přímo na ESB nebo Poskytovateli a Konzumentovi služeb s možností do řídících zpráv přidat identifikátory klíčů a atributy podpisu (certificate chain podepisujícího, identifikátory algoritmu).
- Konfigurace přenosů souborů v grafickém prostředí.
- Možnost rozšířit řešení o funkcionalitu pořizující časová razítka pro soubory a přenášené zprávy.

Podpora řízení ETL úloh

- Podpora řízení ETL úloh pro synchronizaci velkých objemů dat mezi systémy.
- Možnost spouštět ETL úlohy a poskytovat služby Konzumentům pro spouštění ETL úloh.
- Podpora přenosu zpráv o výsledcích ETL úloh.

Zveřejnění služeb a přihlášení k odběru služeb

- Propagace nových služeb do centrálního registru služeb.
- Přihlášení k odběru služeb v centrálním registru služeb.

Transakční zpracování a obsluha chyb

- Podpora transakčního zapracování, transakcí v rámci workflow.
- Schopnost uložit stav transakce v případě nemožnosti kvalitní a včasné odpovědi jednoho nebo více ze zúčastněných koncových bodů zahrnutých v rámci orchestrace služby

a dokončení transakce v okamžiku obnovení dostupnosti koncového bodu (pozastavení a opakované spuštění transakce).

- Zajištění konzistence dat v zúčastněných koncových bodech v případě pozastavení zpracování transakce. Možnost vykonání automatické či manuální kompenzační akce v ovlivněných bodech.
- Schopnost provést roll-back transakce (roll-back i v zúčastněných koncových bodech podporujících transakční zpracování).
- Podpora vlastností ACID (Atomicity , Consistency, Isolation, Durability).
- Podpora pro dlouho běžící procesy a možnost individuálních odbavení transakcí (commit) v jednotlivých zúčastněných systémech.
- Možnost opakovaného volání systémů a konfigurace počtu opakovaných volání (po chybě).
- Přehled běžících transakcí, který bude poskytovat:
    - možnost vyhledat transakci dle kritérií,
    - informace, v jakém stavu se transakce nachází,
    - informace v jakém koncovém bodu, bodech se transakce aktuálně nachází,
    - informace, ve kterém systému transakce selhala,
    - další monitoring informace o transakcích.
- Zajištění proti duplicitě transakcí. Možnost definovat pravidla pro ověření, zda již transakce v systému byla založena či nikoliv (např. zda již byla přijata zpráva příslušného typu ze stejným identifikátorem osoby).
- Možnost nastavení politik pro obsluhu chyb.

Jednotné řízení

- Podpora řízení běhu prostřednictvím událostí (event driven architecture).
- Generování událostí v případě výskytu chyby.
- Zaručené doručení událostí – podpora „store and forward“ funkcionality.
- Řízení sekvencí událostí.
- Možnost zastavení příjmu všech zpráv anebo zpráv pro individuální frontu, přičemž všechny zprávy obdržené před zastavením příjmu budou doručeny a související transakce dokončeny.
- Podpora řízeného vypnutí systému bez dopadu na rozpracované transakce a konzistenci dat v komunikujících systémech (např. nejprve zastavení příjmu požadavků do vstupních front, dokončení všech rozpracovaných transakcí a poté zastavení systému).
- Podpora uchování stavů transakcí a procesů („state machine“).
- Jednotný monitoring, audit, logování, měření, instalace, konfigurace a deployment integračních a dalších ESB komponent napříč celou IT infrastrukturou.
- Jednotný bezpečnostní management, digitální podpisy a šifrování zpráv.

- Optimalizace služeb podle zátěže.
- Orchestrace služeb, volání zúčastněných koncových bodů v rámci definovaného workflow.
- Schopnost obsluhovat selhání a chyby v rámci workflow.
- Schopnost iniciace workflow v ESB AgriBus na základě externí události.

Quality of service

- Zaručené doručení, doručení právě jednou.
- Možnost řízení priority zpráv dle Poskytovatele služby či dle Konzumenta služby.
- Mechanismy pro zaručení vysoké dostupnosti (high-availibility, fail-over) a přechod do záložních center (disaster recovery).
- Inteligentní a přitom transparentní rozložení zátěže pro zaručení škálovatelnosti (load balancing).
- Nastavení kvality komunikace (timeouty, priorita, způsob přenosu, řízení objemu komunikace) a ochrana koncových prvků před nadměrnou zátěží.

3.4.2 Procesní platforma AgriBus BPM

Řešení AgriBus musí poskytovat funkcionality pro podporu běhu dlouho trvajících procesů zahrnujících lidské interakce. Mezi základní požadované funkcionality patří:

- podpora dlouho trvajících procesů,
- možnost návrhu procesů v grafickém prostředí a zároveň přímou editací definičních souborů,
- podpora návrhu, běhu a zobrazení webových formulářů zprostředkujících uživatelské vstupy v rámci procesů,
- možnost rozšířit standardní funkcionality webových formulářů o nově vyvinuté aplikační komponenty a nové komponenty grafického uživatelského rozhraní,
- možnost přidávat další procesní aplikace a agendy bez negativních dopadů na provoz řešení AgriBus a zejména na komunikační vrstvu ESB,
- podpora integrace webových rozhraní procesů a aplikací do portálového řešení Objednatele, zejména podpora Single-sign-on a ověřování oproti LDAP Objednatele a podpora Java Portlet Specification,
- podpora přístupu prostřednictvím mobilních platforem (zejména Android, IOS),
- podpora automatické aktualizace dat ve formulářích či aktualizace na žádost v případě změny podkladových zdrojových dat.

4 ESB - Enterprise service bus

ESB je softwarové řešení (architektura), která umožňujme komunikaci mezi více spravovanými systémy. Základním konceptem ESB je snížení počtu propojení mezi jednotlivými aplikacemi a zavedení přidaných hodnot do této komunikace.

ESB spojuje a zprostředkovává všechny komunikace a interakce mezi službami. Zároveň dovoluje služby a procesy rychle měnit, snadno je připojovat, zviditelnit a řídit.

ESB vyznačuje těsnou integrací s aplikacemi. Jednotlivé funkční bloky jsou distribuovány podle potřeby tak, aby spolupracovaly v harmonii a vytvářely tak celkový uživatelský obraz z jediné sběrnice. Umístění ESB blízko jednotlivým aplikacím získáme homogenní prostředí, které jednotným způsobem zajišťuje bezpečnost, audit, dohled a řešení chybových stavů.

4.1.1 Typy spolupráce mezi SOA služby:

Kooperace

Služba využívá prostředky jiné (rovnocenné) služby pro realizaci nabízených funkcí.

Agregace

Služba sestavená ze dvou (nebo více, podřízených) služeb nabízí kombinaci funkcí dílčích služeb.

Choreografie

Služby potom spolupracují za účelem provedení konkrétního business procesu.

Orchestrace

Služba řídí součinnost ostatních služeb za účelem provedení své části business procesu.

4.1.2 Základní integrační principy

V této kapitole popíšeme pět základních integračních principů:

Orchestrace

Složení několika existujících jednodušších komponent do jedné vysokoúrovňové služby. Tím získáme lepší granularitu služby a podpoříme znuvupoužitelnost jednodušších služeb.

Transformace

Transformování dat mezi kanonickým formátem a specifickým formátem dat vyžadovaným ESB konektorem. Jako příklad může posloužit transformace mezi CSV, Cobol copybook nebo EDI formátem do SOAP/XML nebo JSON. Kanonický datový formát zjednodušuje transformační

požadavky u velkých ESB implementací, kde je více poskytovatelů a spotřebitelů, kde každý z nich používá své vlastní formátování data a jiných definic.

Transponace

Transportní protokol je vybrán z více podporovaných formátů. (Např.: HTTP, JMS, JDBC).

Mediace

Poskytnutí více rozhraní za účelem:

- podpora více verzí služby pro zachování zpětné kompatibility,
- poskytnutí více kanálů pro poskytnutí stejné komponenty, to se například využívá, pokud je třeba ke stejné komponentě přistupovat se starším a novějším rozhraním.

Nefunkciální konzistence

Sjednocení způsobu, kterým je prováděna bezpečnostní a monitorovací politika. Dalším cílem je vylepšení dostupnosti služeb využíváním vícenásobnou instancí ESB, zvýšením její propustnosti a odstraněním jednobodových rizikových míst systému.

5 Porovnání Open source ESB

5.1 Srovnávací kritéria

- Komunita - má produkt aktivní fórum nebo e-mailové konference. Jsou k nalezení články, návody nebo video návody.
- Dostupnost placené podpory - zda je možné sehnat komerční podporu pro vybrané ESB.
- Podpora BPM – zda a jakou BPM platformu lze s vybranou ESB používat.
- BPEL - zda ESB umožňuje zpracovávat BPEL.
- Monitoring - zda a jaká existuje utilita pro monitorování a správu provozu.
- Podporované OS - na kterých OS lze vybranou ESB používat.
- Živost vývoje - zde se podíváme na datum vydání poslední verze.
- Licence - pod jakou licencí se daná ESB platforma vydává.
- Cena - zda lze produkt používat zdarma.

5.2 Představení porovnávaných ESB

JBoss Fuse

Webové stránky projektu: http://www.jboss.org/why-subscribe/

JBoss je vytvářen firmou RedHat. Hlavní jeho výhodou je dobrá dostupnost podpory a velké množství návodů a video tutoriálů. Lze také rozšířit o Boss BPM Suite pro správu business procesů a o BPEL designer pro modelování procesů. Podporované operační systémy jsou Linux a Windows server. JBoss je k dispozici pod licencí Apache. Pro vývojáře je k dispozici zdarma, jinak se platí cena 8 000 $ na 16 jader.

Apache ServiceMix

Apache ServiceMix lze nalézt na webu: http://servicemix.apache.org/

Apache ServiceMix podporuje BPM engin Activiti. Podporu tohoto projektu poskytuje open-source komunita, ale existuje také firma Ameliant, která poskytuje komerční podporu.

MuleSoft – ESB Solutions

MuleSoft ESB je k dispozici https://www.mulesoft.com/platform/soa/mule-esb-open-source-esb

Ke stažení nabízí pouze 30denní verzi. Jinak se jedná o placený produkt. MuleSoft taktéž vyvíjí BPM nástroj.

WSO2

WSO2 je open-source projekt, který je i pro komerční použití zdarma. Platí se pouze případná enterprice-podpora a nabízené kurzy. Komunitní podpora využívá k diskuzi server Stack-overflow, ale na mnoho dotazů je zde aktuálně bez odpovědi. Pro podporu BPM obsahuje grafický editor služeb. Stránka projektu je: http://wso2.com/products/enterprise-service-bus/

Open ESB

Open ESB je k dispozici na http://www.open-esb.net/ OpenESB má poměrně živou komunitu, lze také najít nejrůznější video návody. Open ESB nenabízí nástroj pro BPM řešení.

Spring Integration

V případě Spring Integration se jedná pouze o programový Framework, který je navržen pro podporu integrace. Více informací o tomto projektu lze nalézt na adrese: http://projects.spring.io/spring-integration/

Petals ESB

Petals ESB je vyvíjené společností OW2 Middleware Consortium a je dostupné na webové adrese http://petals.ow2.org/ Pro správu BPM umožňuje integraci vlastního Bpm editoru, který je možné vyzkoušet na adrese: http://bpmneditor.petalslink.com/

Talend ESB

Dalším porovnávaným je Talend ESB, který je k nalezení na: https://www.talend.com/resource/open-source-esb.html. Talend ESB je navržen pro podporu zpracování BigData.

Oracle SOA Suite

Výběr představovaných open source ESB ukončujeme uzavřenou implementací od Oracle. SOA Suite bylo zahrnuto do porovnávací tabulky z důvodu porovnání funkcionality vůči open source řešení, a také z důvodu, že z této platformy budeme migrovat stávající řešení.

5.3 Porovnání vybraných ESB

Pro vybrané ESB jsme odpověděli na porovnávací kritéria, definovaná výše. Odpovědi na tyto kritéria jsme vyplnili do tabulky pro snadnější porovnání.

		JBoss – Fuse	Apache Service Mix	MuleSoft - ESB	WSO2
Podpora	**Enterprise support**	Ano	Ano	Ano	Ano
	Návody	Ano	ano	ano	Ano
	Fórum	Aktivní	Aktivní	Aktivní	Aktivní
Udržovatelnost	**Podporované BPM**	JBoss BPM	Activiti	MuleSoft BPM	WSO2 Business Process Server
	integrace s BPEL	BPEL Designer	Apache ODE	Ano	WS-bpel
	Monitoring tools	JBoss Operations Network	JMX	Mule Enterprise Management	JMX, a další nástroje pro statistiku
Platba	**Cena**	Zdarma pro vývojáře, placeno při komerčním použití	Zdarma	Placený produkt	Zdarma
	Licence	Apache	Apache License 2.0	CPAL 1.0	Apache License, Version 2.
	OS support	Red Hat Enterprise Linux 7-5 Microsoft Windows Server 2012	Windows Linux	Windows Linux	Windows Linux
Živost projektu	**Datum vydání poslední verze ke dni 10.12.2015**	23.06.2015	24.11.2015	15.01.2015	02.12.2015

		Open ESB	Spring Integration	Petals ESB	Talend ESB	Oracle SOA Suite
Podpora	**Enterprise Support**	Ano	Ano	Ano	Ano	Ano
	Návody	Ano	ano	ano	Ano	Ano
	Fórum	Ano	neaktivní	neaktivní	aktivní	Aktivní
Udržovatelnost	**Podporované BPM**	-	Ne vlastní, ale lze Activiti	BMP editor	-	Oracle BPM
	integrace s BPEL	Ano	Ano, lze použít Oracle BPEL Process Manager	Ano	Ne	Oracle BPEL Process Manager
	Monitoring tools	OpenESB Monitor	JMX	Cacti	Service Activity Monitoring	Oracle Business Activity Monitoring
Platba	**Cena**	Zdarma	Zdarma	Zdarma	Zdarma	Placený produkt
	Licence	CDDL 1.0	Apache License 2.0	Lesser General Public License 2.1	Apache open source license	Proprietární
	OS support	Windows Linux	(Java SE 7,8)	Windows Linux	Windows Linux	Windows Linux
Živost projektu	**Datum vydání poslední verze ke dni 10.12.2015**	07.05.2015	09.12.2015	01.10.2015	11.12.2014	26.10.2015

5.4 Výběr ESB

Dle tabulek z předchozí kapitoly jsme mohli vybrat ESB k detailnímu porovnání. Dle tabulky jsme vyřadili Spring Integration a Petals ESB pro méně aktivní komunitní fórum, než jaké je u ostatních řešení.

Dále jsme vyřadili OpenESB a Talend ESB, protože neumožňují snadnou integraci BPM modulu. Ostatní řešení mají podporované BPM, které lze použít.

Z důvodu cenové politiky jsme se rozhodli vyřadit JBoss – Fuse a MuleSoft. Protože cílem práce je pouze ukázání možnosti komunikace, rozhodli jsme se preferovat řešení použitelné zdarma.

Zadaným požadavkům tak vyhověl Apache ServiceMix, WSO2 a Oracle SOA Suite. Vybráním kterékoliv z těchto řešení bychom neudělali chybu. Oracle SOA Suite jsme nevybrali, protože se nejedná o Open Source řešení. Do našeho porovnání jsme jej zařadili pro porovnání možností open source a proprietárních řešení.

Rozhodnutí padlo na Apache ServiceMix. Podporuje všechna porovnávací kritéria a vývojářská komunita se může pyšnit i kvalitní dokumentací.

5.5 Porovnání požadavků ESB Agribus s možnostmi ESB Apache ServiceMix

Podpora transportů:

- http,
- JMS,
- JCA,
- JavaBean.

Podpora webových služeb:

- Podpora SOAP 1.1 s využitím JAX-RPC,
- Podpora SOAP 1.1/JMS,
- Podpora SOAP 1.2.

Podpora standardizovaných adaptérů a konektorů v podobě SCA komponent:

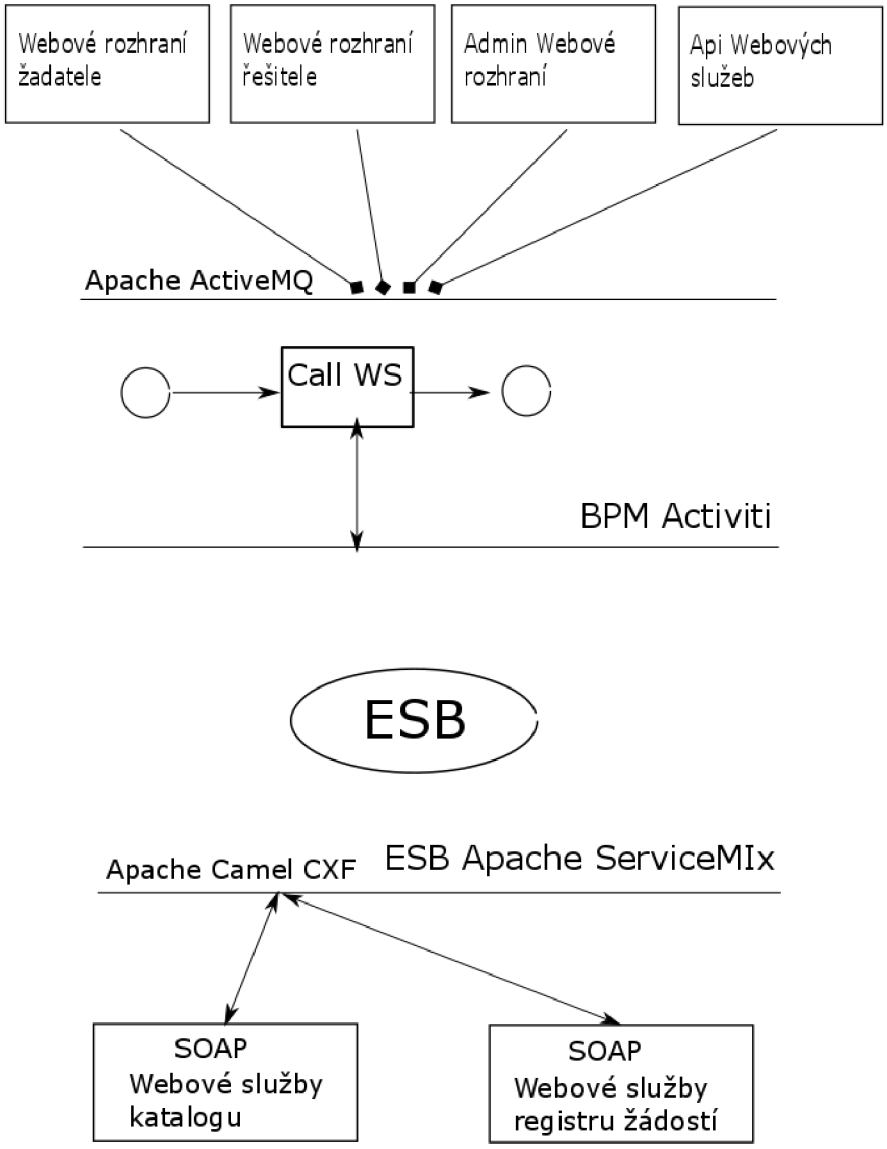
- Email,
- Sftp,
- JDBC.

Podpora autentizace s využitím LDAP

Zvolené ESB Apache ServiceMix splňuje všechny výše vyjmenované zkoumané požadavky, proto jej můžeme použít pro implementaci ESB Agribus. Informace nalezeny dle zdroje [3].

6 Plán nasazení vybraného ESB

Z informací předložených v předchozích kapitolách nyní můžeme vytvořit schéma komunikace jednotlivých komponent. Cílem této kapitoly je vytvořit plán, který povede k nainstalování a propojení vybraných komponent – Apache ServiceMix, Activiti BPM, Webové služby ISND a uživatelského webového rozhraní.



Obrázek 10 - Plánované propojení komponent systému

6.1 Instalace virtuálního serveru

K instalaci virtuálního serveru vybereme a nainstalujeme vhodnou linuxovou distribuci, která dokáže hostit všechny požadované služby. My jsme vybrali distribuci Lubuntu 14.04 LST. Tato distribuce byla zvolena pro své nízké paměťové nároky. Je proto vhodná pro běh ve virtuálním prostředí, které budeme využívat. Virtualizovat budeme pomocí aplikace VirtualBox.

6.2 Instalace ESB – Apache ServiceMix

Nainstalujeme a zprovozníme zvolené řešení ESB – Apache ServiceMix 6.1. ServiceMix se stahuje jako spustitelná aplikace. Ke stažení je k dispozici na webových stránkách dle zdroje [3]. Ke spuštění je potřeba mít nainstalováno Java Runtime Environment (JRE) 1.6. nebo novější a nastavenou systémovou proměnnou JAVA_HOME.

6.2.1 Instalace vybraných komponent

Komponenty rozšiřují poskytované rozhraní ESB sběrnice. V našem případě budeme potřebovat zprovoznit komunikaci s webovými služby prostřednictvím protokolu SOAP a vybrat způsob pro komunikaci s uživatelským webovým rozhraním.

Apache Camel

Apache Camel je Framework vytvořený v Jave, který se snaží zjednodušovat integraci. Obsahuje implementaci všech běžně používaných Enterprise Integration Patterns (EIP), umožňuje přeposílání a transformaci zpráv mezi velkým množstvím typů prvků.

Apache CXF

Apache CXF je komponenta, která umožňuje vytvářet a používat rozhraní služeb JAX-WS a JAX-RS. Tyto služby pak přináší podporu komunikace pomocí protokolů SOAP, XML / HTTP, RESTful HTTP a CORBA a práci s řadou transportů jako HTTP, JMS nebo JBI.

Apache ActiveMQ

Apache ActiveMQ je rozhraní pro zprostředkování zpráv. ActiveMQ je naimplementováno v Jave a nabízí plnou podporu JMS (Java Message Server). ActiveMQ umožňuje přeposílání zpráv napříč různými klienty a protokoly. My jej využijeme pro komunikaci uživatelského rozhraní s Apache Camel.

6.3 Instalace Activiti BPM

Activiti BPM je jednoduché workflow a Business Proces Management (BPM). Platforma je zaměřená na obchodníky, vývojáře a systémové administrátory. Activiti využívá BPMN2 proces engine pro Javu. Je jednoduše integrovatelné do jakékoliv Java aplikace. Jedná se o open source nástroj distribuovaný pod Apache licencí. Activiti BPM Engin lze nainstalovat jako další komponentu Apache ServiceMix. [4]

6.4 Simulace ISND

Pro zvolené služby bude potřeba navrhnout komunikační protokol WSDL. Pro jejich simulaci budeme používat SOAP UI. Budeme simulovat některé z komponent spadající pod Katalog dotačních programů a Registr žádostí.

6.5 Napojení ISND na ESB

Webové služby ISND budeme volat pomocí Apache Camel. Apache Camel umožňuje definici Camel rout, jedná se o popis směrovacích pravidel pomocí Java DSL (Domain Specific Language). Camel umožňuje definovat vstupní a výstupní bod a transformační logiku mezi nimi. My budeme využívat volání webové služby ISND pomocí komponenty Apache CXF.

6.6 Ověření funkčnosti komunikace

Na jednoduchém procesu ukážeme funkčnost celé komunikace instalovaných komponent. Tím ukážeme, že Informační systém Národní dotace 2016, je připraven na budoucí nasazení v rámci TO-BE Architektury Agribus.

7 Simulace ISND

Informační systém Národní dotace v současném stavu je implementován jako webová aplikace v .NET s využitím návrhového vzoru mvc. V současném stavu tedy ISND není přímo připraven pro integraci s ESB Agribus. Přesto díky vhodnému návrhu nebude složité vybrané kontrolery předělat na webové služby, které tuto integraci umožní.

V této kapitole popíšeme tvorbu wsdl rozhraní webové služby Formální kontrola, která bude součástí komponenty ISND – Registr žádostí. Formální kontrola slouží k validaci podaných formulářů. Tato služba kontroluje, zda jsou vyplněna všechna povinná pole, zda všechny pole obsahují přípustné hodnoty, zda jsou podány potřebné přílohy a splněny další podmínky dle dotačního programu. V naší ukázce vytvoříme rozhraní wsdl webové služby pro kontrolu formuláře žádosti.

7.1 Tvorba wsdl služby formální kontrola

1) **`wsdl:definitions`** Kořenová značka musí mimo standartní atributy obsahovat jméno služby a definici jmenného prostoru.

```
<wsdl:definitions name="ISND"
                  targetNamespace="http://vut.pds/isndNamespace/"
                  xmlns:ns1="http://schemas.xmlsoap.org/soap/http"
                  xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
                  xmlns:tns="http://vut.pds/isndNamespace/"
                  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
                  xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/">
```

2) **`wsdl:types`** Značka, která bude obsahovat definici typů pro přijímání a odesílání zpráv všech služeb. Volaná služba `registrZadostiIsValid` bude přijímat data z dotačního formuláře. Služba bude vracet odpověď `registrZadostiIsValidResponse`, která bude informovat o výsledku formální kontroly zvoleného formuláře.

```
<wsdl:types>
    <xs:schema elementFormDefault="unqualified"
               targetNamespace="http://vut.pds/isndNamespace/"
               version="1.0"
               xmlns:xs="http://www.w3.org/2001/XMLSchema">
      <xs:element name="registrZadostiIsValid"
                  type="tns:registrZadostiIsValid"/>
      <xs:element name="registrZadostiIsValidResponse"
                  type="tns:registrZadostiIsValidResponse"/>
      <xs:complexType name="registrZadostiIsValid">
        <xs:sequence>
          <xs:element name="dotProgram" type="xs:string"/>
          <xs:element name="dotPredmet" type="xs:string"/>
          <xs:element name="Pocet" type="xs:int"/>
          <xs:element name="Sazba" type="xs:int"/>
          <xs:element name="Pozadovano" type="xs:int"/>
        </xs:sequence>
      </xs:complexType>
      <xs:complexType name="registrZadostiIsValidResponse">
        <xs:sequence>
          <xs:element name="return" type="xs:string"/>
        </xs:sequence>
      </xs:complexType>
    </xs:schema>
  </wsdl:types>
```

3) **`wsdl:message`** Abstraktní typ definující, jaká data se budou přeposílat. K definici využijeme definicí vytvořených ve `wsdl:types`.

```
<wsdl:message name="registrZadostiIsValid">
    <wsdl:part name="parameters"
               element="tns:registrZadostiIsValid">
    </wsdl:part>
</wsdl:message>
    <wsdl:message name="registrZadostiIsValidResponse">
    <wsdl:part name="parameters"
               element="tns:registrZadostiIsValidResponse">
    </wsdl:part>
</wsdl:message>
```

4) **`wsdl:portType`** Popíšeme množinu operací poskytovaných definovanou službou.

```
<wsdl:portType name="ISND">
    <wsdl:operation name="registrZadostiIsValid">
      <wsdl:input name="registrZadostiIsValid"
                  message="tns:registrZadostiIsValid">
      </wsdl:input>
      <wsdl:output name="registrZadostiIsValidResponse"
                   message="tns:registrZadostiIsValidResponse">
      </wsdl:output>
    </wsdl:operation>
</wsdl:portType>
```

5) **wsdl:binding** Pro popisovaný typ portu popíšeme konkrétní protokol a formát dat.

```
<wsdl:binding name="ISNDSoapBinding" type="tns:ISND">
    <soap:binding style="document"
                  transport="http://schemas.xmlsoap.org/soap/http"/>
    <wsdl:operation name="registrZadostiIsValid">
      <soap:operation soapAction="" style="document"/>
      <wsdl:input name="registrZadostiIsValid">
        <soap:body use="literal"/>
      </wsdl:input>
      <wsdl:output name="registrZadostiIsValidResponse">
        <soap:body use="literal"/>
      </wsdl:output>
    </wsdl:operation>
</wsdl:binding>
```

6) **wsdl:wervice** Množina souvisejících koncových bodů.

```
<wsdl:service name="ISND">
    <wsdl:port name="isndPort" binding="tns:ISNDSoapBinding">
      <soap:address
location="http://martin-VirtualBox:8088/mockISNDSoapBinding"/>
    </wsdl:port>
</wsdl:service>
```

Pro kontrolu a ověření správnosti vytvořeného wsdl můžeme využít online validátor dostupný na webové adrese https://www.wsdl-analyzer.com/.

7.2 Simulace webové služby

Webovou službu formální kontrola, jejíž rozhraní jsme v předchozí kapitole nadefinovali pomocí wsdl, budeme simulovat pomocí programu SoapUI. SoapUI je open source aplikace k testování servisně orientovaných architektur.

Vytvořené wsdl do programu SoapUI naimportuje a vygeneruje soap mockup servis, který bude simulovat běh webové služby. Aby simulace vracela různé výsledky, vytvoříme dvě různé odpovědi s návratovou hodnotou `True` a `False`. Simulovaná webová služby nyní bude tyto odpovědi střídavě vracet. K ověření funkčnosti webové služby můžeme použít vygenerovaný dotaz prostřednictvím SoapUI. Vytvořené Wsdl nyní bude přístupné i přes http na adrese: http://localhost:8088/mockISNDSoapBinding?WSDL

8 Implementace

8.1 Apache ServiceMix, Apache Camel, Activiti BPM - zavolání webové služby

V této kapitole ukážeme integraci pomocí Apache ServiceMix. Pomocí tohoto integračního kontejneru ukážeme spolupráci Activiti enginu a Apache Camel. Popíšeme instalaci všech potřebných komponent a jejich spolupráci ukážeme na jednoduchém příkladu, který bude demonstrovat použití webového rozhraní ISND v ESB Agribus.

Cílem tohoto příkladu bude vytvořit business proces řízený pomocí Activiti. Tento proces se bude spouštět vložením souboru do vybrané složky. První komponenta business procesu obsah tohoto souboru využije k zavolání předem definované webové služby. Z důvodu, že cílem je ukázání možnosti a principu spolupráce Activiti a Apache Camelu, nebudeme tuto webovou službu volat přímo pomocí `WebServiceTask` z Activiti, ale vytvoříme vlastní komponentu popsanou Camelem, která toto zavolání provede. Touto kombinací získáme daleko větší spektrum možných dalších integrací a transformací využitelných k řízení business procesu.

8.1.1 Tvorba projektu

Základní strukturu projektu vygenerujeme pomocí Apache Maven. Maven je nástroj pro tvorbu a překlad Java projektů. Maven umožňuje jednoduše vyřešit sestavení a závislosti projektu. V našem případě budeme používat současnou aktuální verzi 3.3.9. Se starší verzí Maven nepůjdou některé součásti projektu přeložit.

Konfigurační soubor

Konfigurace použitých balíčků se provádí v souboru pom.xml. Pro možnost nasazení projektu pomocí Apache ServiceMix je potřeba nastavit atribut packaging na bundle.

- **Je potřeba nadefinovat závislosti na potřebné projekty:**
  - **`Camel-core`** – základní modul Apache Camel.
  - **`Camel-cxf`** – umožňuje používat rozhraní webových služeb. V tomto příkladu bude použito pro komunikaci s webovou službu prostřednictvím protokolu Soap.
  - **`Activiti-camel`** – tento balíček umožňuje popisovat chování komponent bpm procesu pomocí Camel rout.

- **Definice plugin:**
  - **Cxf-codegen-plugin** – tento plugin použijeme ke generování Java kódu z wsdl, které jsme vytvořili. Díky tomu budeme moct z Camelu komunikovat s vystavenou webovou službou.

```
<plugin>
    <groupId>org.apache.cxf</groupId>
    <artifactId>cxf-codegen-plugin</artifactId>
    <executions>
        <execution>
            <id>generate-sources</id>
            <phase>generate-sources</phase>
            <configuration>
                <sourceRoot>
                ${basedir}/target/generated/src/main/java
                </sourceRoot>
                <wsdlOptions>
                    <wsdlOption>
                        <wsdl>
           http://localhost:8088/mockISNDSoapBinding?WSDL
                        </wsdl>
                    </wsdlOption>
                </wsdlOptions>
            </configuration>
            <goals>
                <goal>wsdl2java</goal>
            </goals>
        </execution>
    </executions>
</plugin>
```

  - **Maven-bundle-plugin** – tento plugin slouží ke generování jar balíčku.

```
<plugin>
    <groupId>org.apache.felix</groupId>
    <artifactId>maven-bundle-plugin</artifactId>
    <version>2.4.0</version>
    <extensions>true</extensions>
    <configuration>
        <instructions>
            <Bundle-SymbolicName>
                ${project.artifactId}
            </Bundle-SymbolicName>
            <Bundle-Version>
                ${project.version}
            </Bundle-Version>
            <Export-Package>
                edu.vut.test*;version=${project.version}
            </Export-Package>
            <Import-Package>*</Import-Package>
        </instructions>
    </configuration>
</plugin>
```

8.1.2 OSGi Blueprint

Blueprint je implementací standartu OSGi, který slouží k popisu integrace. Je to jeden ze způsobů popisu Camel rout. Camel routa umožňuje popis vstupního rozhraní pro zprávy, jejich transformace, filtrování a přeposlání na výstupních rozhraní.

V našem příkladu jej využijeme pro definici `cxf-endpointu`, který budeme v Camel routách používat. `Cxf-endpoint` je definovaným odkazem na webovou službu a také jmenným prostorem, do kterého plugin wsdl2java vygeneroval obslužné třídy.

```
<cxf:cxfEndpoint id="myWebService"
            address="http://localhost:8088/mockISNDSoapBinding"
            serviceName="s:ISND"
            endpointName="s:isndPort"
            wsdlURL="http://localhost:8088/mockISNDSoapBinding?WSDL"
            xmlns:s="http://vut.pds/isndNamespace/">
    <cxf:properties>
        <entry key="dataFormat"
               value="PAYLOAD"/>
        <entry key="defaultOperationNamespace"
               value="http://vut.pds/isndNamespace/"/>
        <entry key="defaultOperationName"
               value="registrZadostiIsValid"/>
    </cxf:properties>
</cxf:cxfEndpoint>
```

Dále použijeme `CamelContext` k nadefinování Camel rout, které budou interagovat s Activiti:

```
<camelContext id="camelContext"
              xmlns="http://camel.apache.org/schema/blueprint">
    <packageScan>
        <package>org.apache.servicemix.examples.activiti</package>
    </packageScan>
</camelContext>


<service interface="org.activiti.camel.ContextProvider">
    <bean class="org.activiti.camel.SimpleContextProvider">
        <argument value="OrderProcess"/>
        <argument ref="camelContext"/>
    </bean>
</service>
```

Vytvořený ContextProvider umožní komponentě business procesu zavolat Camel routu pomocí výrazu `activiti:delegateExpression="${camel}"`

8.1.3 Business proces

Popíšeme business proces. Proces bude tvořen počátečním a koncovým uzlem a dvěma komponentami. Proces budeme spouštět využitím Camelu, který bude popsaný níže. První komponenta `processOrder` bude sloužit k zavolání webové služby. Druhá komponenta `processDelivery` bude sloužit k výpisu zprávy získané z webové služby. [5]

Obrázek 11 - Business proces 1

```
<process id="OrderProcess" isExecutable="true">
    <startEvent id="start"/>
    <sequenceFlow id="flow1" sourceRef="start" targetRef="processOrder"/>
    <serviceTask id="processOrder" ctiviti:delegateExpression="${camel}"/>
    <sequenceFlow id="flow2" sourceRef="processOrder"
                  targetRef="processDelivery"/>
    <serviceTask id="processDelivery"
                 activiti:delegateExpression="${camel}"/>
    <sequenceFlow id="flow4" sourceRef="processDelivery" targetRef="end"/>
    <endEvent id="end"/>
</process>
```

8.1.4 Definice Camel rout

Pro definici Camel rout vytvoříme třídu `ActivitiRouteBuilder`, kterou vytvoříme rozšířením z třídy `RouteBuilder`. V této třídě přepíšeme implementaci metody `configure`, ve které nadefinujeme chování Camel rout. [6]

Spuštění byznys procesu

Byznys proces bude spouštěn po vložení souboru do vybrané složky. Procesu musíme nastavit property `body`, které obsahuje třídu s metodou `getProcessVariables`. Existence této metody je důležitá pro pokračování procesu s využitím Activiti enginu. Dále procesu nastavujeme identifikátor procesu do `PROCESS_KEY_PROPERTY`.

Business proces spustíme přeposláním na endpoint `activiti:OrderProcess`.

```
from("file:var/activiti-camel/order")
    .setBody(bean(procesInstance))
    .setProperty(PROCESS_KEY_PROPERTY, simple("file:name"))
    .to("activiti:OrderProcess");
```

Zavolání webové služby

V tomto odstavci popíšeme, jak zavolat nadefinovanou webovou službu z komponenty procesu typu `serviceTask` implementované s využitím Apache Camel. Komponenta se spouští tím, že se proces dostane do stavu `processOrder`. Při zavolání použijeme parametr `copyVariablesTo-Properties`, čímž si zpřístupníme proměnné uložené v proces body metodou `getProcess-Variables`.

Atribut body před zavoláním webové služby nahradíme obsahem souboru, kterým jsme proces spustili. V tomto případě předpokládáme, že obsah souboru bude mít správný formát. Samotnou webovou službu zavoláme pomocí `cxf:endpointu`, který jsme nadefinovali v blueprintu:

```
to("cxf:bean:myWebService")
```

Po zavolání atribut `body` bude obsahovat zprávu vrácenou z webové služby. Pro správné pokračování procesu prostřednictvím Activiti enginu musíme hodnotu `body` přepsat použitím metody `setBody`, tentokrát s parametrem `bean(procesDeliveryInstance)`. Tato třída je obdoba třídy `ProcesInstance` použité v první routě. Rozdíl je v nastavení parametru `msg`, který nyní bude přebírat aktuální `body`, tedy návratovou zprávu z webové služby na místo obsahu souboru.

V závěru routy proces bude pokračovat komponentou `receiveDelivery`.

```
from("activiti:OrderProcess:processOrder?copyVariablesToProperties=true")
    //Nastavime body na obsah souboru
    .setBody(simple("${property.message}"))
    .setProperty(PROCESS_KEY_PROPERTY, simple("${property.orderid}"))
    //Zavolani webove sluzby
    .to("cxf:bean:myWebService")
    //Znovu nastaveni body na tridu obsahujici getProcessVariables
    .setBody(bean(myHelper))
    .to("activiti:OrderProcess:receiveDelivery");
```

Vypsání přijaté zprávy

Zprávu vypíšeme v komponentě procesu. V této komponentě opět vykopírujeme proměnné procesu do `properties`, které využijeme k vypsání přijaté zprávy do logu.

```
from("activiti:OrderProcess:processDelivery?copyVariablesToProperties=true
")
    .log("Proces delivery - message ${property.message}")
```

8.1.5 Nasazení a otestování projektu

Spustíme Apache ServiceMix 6.1. Následující příkazy pro ovládání pomocí konzole jsou platné pro tuto verzi.

Pro spouštění námi vytvořeného projektu je potřeba zkontrolovat a doinstalovat potřebné `features`. Jejich seznam, včetně informací, zda jsou nainstalované, získáme příkazem `feature:list`. Pro náš příklad je potřeba mít nainstalované camel-core, camel-blueprint, camel-cxf a activiti. Camel-core a camel-blueprint jsou základní součástí ServiceMixu, ostatní je nutno nainstalovat příkazem `feature:install [název vybrané komponety]`.

Náš projekt nasadíme nakopírováním přeloženého balíčku projektu do složky [ServiceMixHome]/deploy. K ověření správnosti nasazení můžeme použít příkaz `log:display`. Tímto příkazem můžeme vyčíst případné chyby, které se vyskytly při nasazování. Pokud vše proběhlo v pořádku, nalezneme náš příklad v seznamu balíčků, který získáme příkazem `bundle:list`.

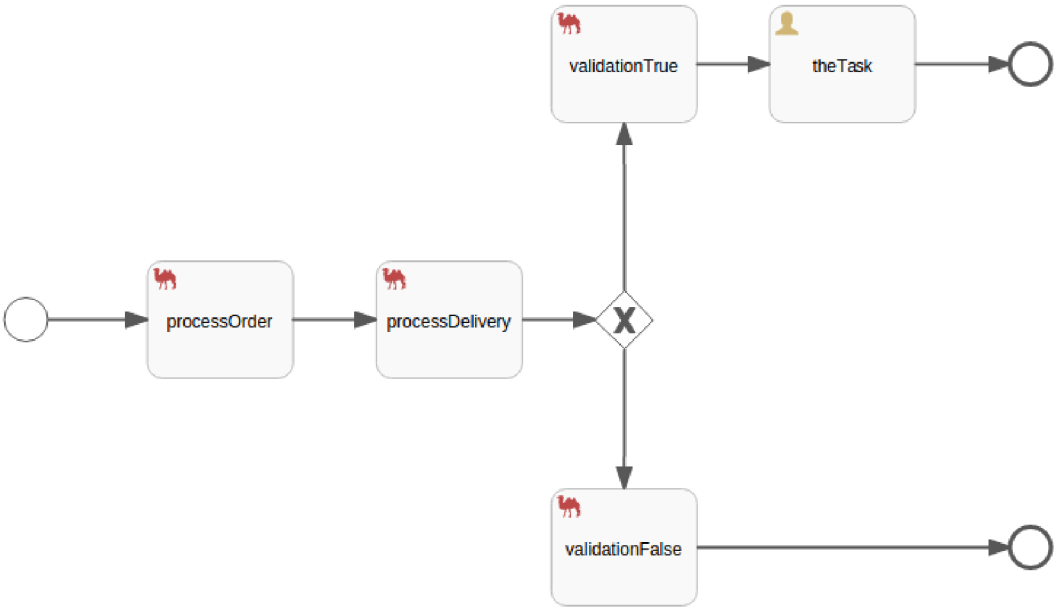
Otestování funkčnosti příkazu provedeme vložením souboru do složky ${ServiceMixHome}/var/activiti/order. Obsah souboru se bude používat při zavolání webové služby, musí mít proto následující strukturu:

```
<?xml version="1.0" encoding="UTF-8"?>
<ns1:registrZadostiIsValid xmlns:ns1="http://vut.pds/isndNamespace/">
    <dotProgram>1.D</dotProgram>
    <dotPredmet>1.D</dotPredmet>
    <Pocet>6</Pocet>
    <Sazba>180</Sazba>
    <Pozadovano>1080</Pozadovano>
</ns1:registrZadostiIsValid>
```

Pokud naše služba funguje správně, soubor bude ze složky order vymazán a pomocí příkazu `log:display` vypíšeme odpověď z webové služby.

8.2 Rozšíření základního příkladu

V následujících kapitolách rozšíříme business proces vytvořený v předchozí kapitole o ukázku zpracování informací přijatých z webové služby, ukážeme větvení procesu na základě této informace a především přidáme uživatelskou interakci prostřednictvím webové aplikace Activiti Explorer. Cílem je ukázat možnosti, která zvolené řešení implementace ESB Agribus umožňuje.



Obrázek 12 - Business proces 2

8.3 Zpracování přijatých informací

V této kapitole ukážeme, jak zpracovávat informace ze vstupního souboru a informace přijaté z webové služby. Tyto informace v následující kapitole využijeme k řízení business procesu a k tvorbě formuláře zobrazeného s použitím komponenty typu `UserTask`.

Zpracování přijatých informací budeme provádět v již vytvořené komponentě `procesOrder`. Při definování Camel routy této komponenty jsme po zavolání webové služby použili příkaz:

```
.setBody(bean(procesInstance))
```

Nyní se na třídu `procesInstance` podíváme trochu podrobněji. Tato třída bude obsahovat metodu `getProcessVariables`. Tato metoda nám umožní definovat proměnné procesu, které budou dále přístupné pomocí `property.nazev_proměnné`.

V tomto případě uložíme do proměnné `message` hodnotu body, která nyní obsahuje obsah souboru, kterým jsme proces spustili. Do proměnné `timestamp` si vložíme časové razítko, informující o datu a času spuštění procesu.

Dále si do samostatných proměnných uložíme extrahované informace z xml elementů vstupního souboru. Pro jejich získání nejprve přijatou zprávu zformátujeme odstraněním bílých znaků. Následně použijeme XPathBuilder, který umožňuje zpracovávat xml zprávy. Metoda `stringResult` slouží k získání samotné hodnoty vybraného xml elementu. Následně pomocí metody `evaluate` tuto hodnotu převede na string. Obdobným způsobem budeme také zpracovávat zprávu z přijaté z webové služby s využitím metody `setBody(bean(receiveMsg))`. I když se může zdát, že `procesBody` přepíšeme instancí nové třídy, ve skutečnosti seznam dostupných proměnných rozšíříme a aktualizujeme. Původně definované proměnné jsou tak stále dostupné, čehož budeme v dalších krocích využívat [5].

```
public final class ProcesInstance {
    @Handler
    public Map getProcessVariables(@Body String body,
            @Simple("${date:now:yyyy-MM-dd kk:mm:ss}") String timestamp)
    {
        Map<String, Object> variables = new HashMap<String, Object>();
        variables.put("message", body);
        variables.put("timestamp", timestamp);

        String XString = body.trim().replaceFirst("^([\\W]+)<","<");

        String dotProgram = XPathBuilder.xpath("//dotProgram")
            .stringResult().evaluate(getContext(), XString, String.class);
        variables.put("dotProgram", dotProgram);

        String dotPredmet = XPathBuilder.xpath("//dotPredmet")
            .stringResult().evaluate(getContext(), XString, String.class);
        variables.put("dotPredmet", dotPredmet);

        String pocet = XPathBuilder.xpath("//Pocet")
            .stringResult().evaluate(getContext(), XString, String.class);
        variables.put("pocet", pocet);

        String sazba = XPathBuilder.xpath("//Sazba")
            .stringResult().evaluate(getContext(), XString, String.class);
        variables.put("sazba", sazba);

        String pozadovano = XPathBuilder.xpath("//Pozadovano")
            .stringResult().evaluate(getContext(), XString, String.class);
        variables.put("pozadovano", pozadovano);

        return variables;
    }
}
```

8.4 Větvení business procesu

V této kapitole ukážeme, jak využít informace přijaté z webové služby k řízení business procesu. Větvení business procesu provedeme pomocí komponenty `excluseiveGateway`. Z této komponenty proces může pokračovat pouze jednou cestou, ze všech následně definovaných `sequenceFlow`. Oproti definici `SequenceFlow` používané dříve zde použijeme parametr `conditionExpression`, který umožňuje definovat podmínku pro vybranou cestu. V podmínce můžeme využívat proměnné procesu, které jsme definovali dříve [4].

```
<exclusiveGateway id="exclusiveGw" name="Exclusive Gateway" />
    <sequenceFlow id="flow5"
                  sourceRef="exclusiveGw"
                  targetRef="validationTrue">
        <conditionExpression
                  xsi:type="tFormalExpression">
            ${response == "True"}
        </conditionExpression>
    </sequenceFlow>
    <sequenceFlow id="flow6"
                  sourceRef="exclusiveGw"
                  targetRef="validationFalse">
        <conditionExpression
                  xsi:type="tFormalExpression">
            ${response != "True"}
        </conditionExpression>
    </sequenceFlow>
```

Díky tomu, že námi implementovaná simulace webové služby `registrZadostiIsValid` navrací střídě odpovědi `True` a `False`, při každém druhém spuštění procesu se dostaneme na komponentu `validationTrue`.

8.5 Activiti Explorer

V této kapitole popíšeme integraci Activiti Exploreru, pomocí kterého budeme moct řídit business proces z uživatelského rozhraní.

Komponenta business procesu vyžadující uživatelskou interakci se nazývá `userTask`. Této komponentě k obvyklým atributům přidáme atribut `activiti:assignee`, který definuje uživatele, kterému bude úloha přidělena. V naší ukázce jsme zvolili defaultního uživatele s administrátorskými právy jmenující se Kermit. Dále je důležitý parametr `extensionElements` a v něm zanořený `activiti:formProperty`. V tomto poli můžeme nadefinovat formulář, který musí uživatel pro splnění úlohy vyplnit.

Ukázka komponenty typu UserTask

```
<userTask id="theTask" name="User Task" activiti:assignee="kermit"
          activiti:candidateUsers="kermit">
    <documentation>
        Formulář přiznáno 1.D Včely
    </documentation>
    <extensionElements>
        <activiti:formProperty id="dotProram" name="Dotační program"
                               variable="dotProgram" writable="false" />
        <activiti:formProperty id="dotPredmet" name="Dotační předmět"
                               variable="dotPredmet" writable="false" />
        <activiti:formProperty id="pocet" name="Počet včelstev"
                               variable="pocet" writable="false" />
        <activiti:formProperty id="sazba" name="Sazba požadováno"
                               variable="sazba" writable="false" />
        <activiti:formProperty id="pozadovano" name="Požadováno"
                               variable="pozadovano" writable="false" />
        <activiti:formProperty id="sazbaPriznano" name="Sazba přiznáno"
                               type="long" readable="true"
                               writable="true" />
        <activiti:formProperty id="celkemPriznano" name="Přiznáno"
                               type="long" readable="true"
                                writable="true"/>
    </extensionElements>
</userTask>
```

Pro zobrazení a interakci s uživatelskými úlohy použijeme Activiti Explorer. Jedná se o webovou aplikaci, která umožňuje návrh a správu business procesů. Tato aplikace je dostupná pomocí .war balíčku. Aplikaci zprovozníme pomocí Apache TomCat. Jedná se o aplikační server, který umožňuje spustit webové Java aplikace. Po spuštění bude Activiti Explorer dostupný ve webovém prohlížeči na adrese http://localhost:8080/activiti-explorer/, pokud jsme při nastavování Apache TomCat použili výchozí port.

Activiti Explorer nyní může používat, ale pracuje s jinou databází než do které ServiceMix ukládá námi vytvořený business proces. Databázi změníme přepsáním konfiguračního souboru WEB-INF/classes/db.properties. Konkrétně zde bude potřeba předefinovat parametr jdbc.url na cestu k databázovému souboru, se kterým pracuje Activiti engine nainstalovaný do ServiceMixu. Dále nastavíme parametr FILE_LOCK=NO; čím zabráníme blokování databázového souboru pro Activiti Engine ze strany Activiti Exploreru.

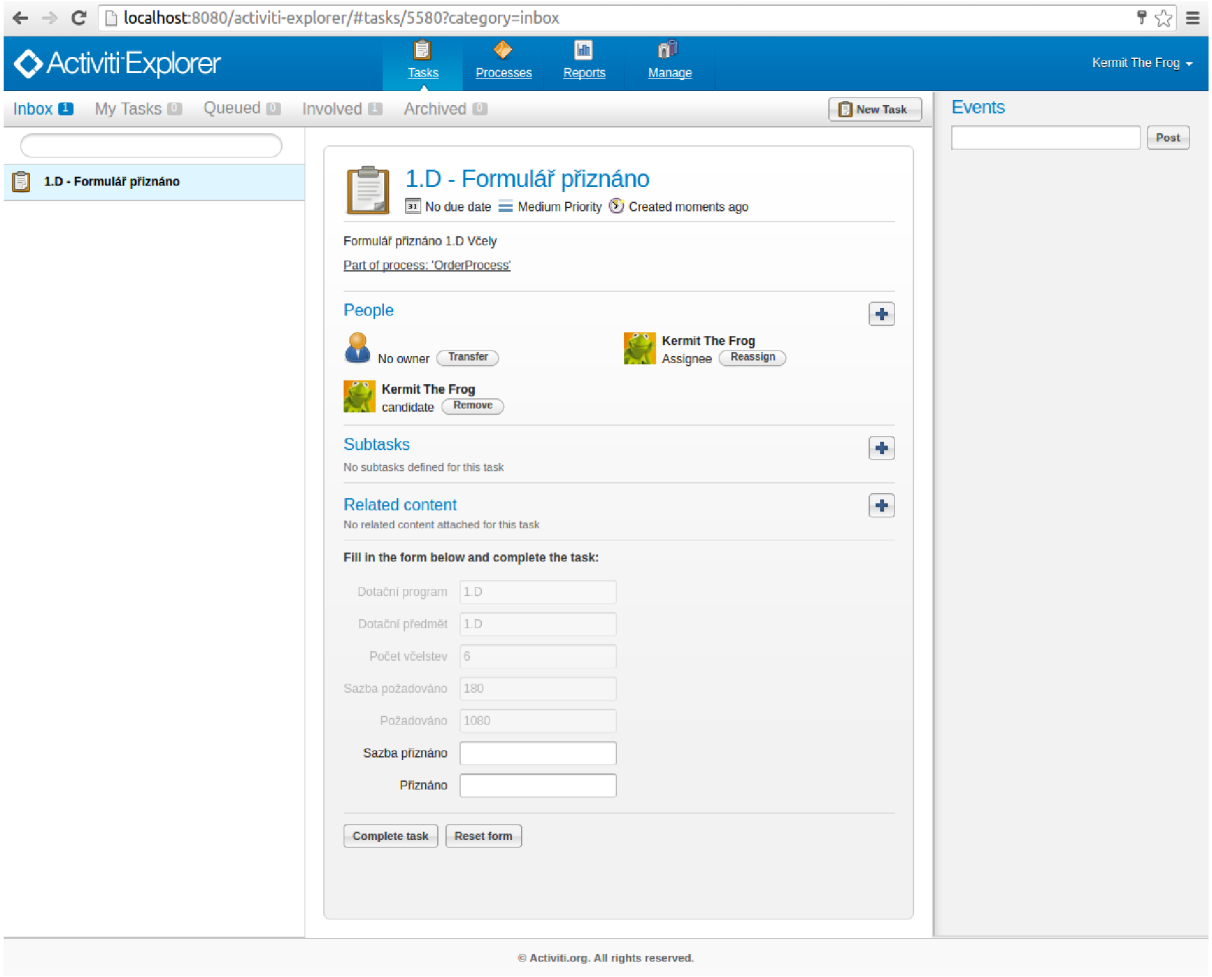
```
jdbc.url=jdbc:h2:file:/home/martin/apache-servicemix-
6.1.0/data/activiti/database;FILE_LOCK=NO
```

V této fázi je potřeba upozornit, že každá verze Activiti enginu pracuje s různým schématem tohoto databázového souboru a také každá verze Activiti Exploreru chce svou podporovanou verzi databázového schéma.

V naší instalaci Apache ServiceMix 6.1.0 se pracuje s Activiti engine 5.19.0. Tato verze pracuje s databázovým schéma verze 5.18.0.1. Proto musíme použít Activiti Explorer 5.19.0. Novější verze nelze použít. Při navázáním databází tímto způsobem je výhodnější nejdříve spustit Activiti

engin prostřednictvím Apache TomCat a pak až následně spustit Apache ServiceMix. Přesto pokud při vývoji vytvoříte ilegální operaci Activiti enginu, může dojít k zablokování databázového souboru a nezbude nic jiného než jej vymazat a vytvořit znovu.

Pokud jsme byli se zprovozněním Activiti Exploreru a napojením na databázi úspěšní, měly by se nám po přihlášení do webového rozhraní pod uživatelem Kermit na záložce Tasks v podzáložce Involved zobrazit uživatelské úlohy pro každou instanci našeho procesu, který se právě nachází na této komponentě. Můžeme zde také spatřit formulář, který jsme nadefinovali v business procesu a také tlačítko CompleteTask. Tímto tlačítkem uživatel dává signál, že úlohu splnil a proces bude pokračovat dále.



Obrázek 13 - Activiti Explorer

8.6 Apache ActiveMQ

V této kapitole ukážeme integraci ActiveMQ rozhraní. ActiveMQ je open source řešení pro zprostředkování zpráv. ActiveMQ je vytvořen v Java a je postaven nad specifikací JMS. Spolu s Apache Camel implementuje Enterprise integration patern (EIP). Z tohoto důvodu je součástí základní instalace Apache ServiceMix.

AcitveMQ budeme využívat ke spouštění business procesu po přijetí zprávy na toto rozhraní. Zprávu budeme odesílat z webového formuláře pomocí php skriptu.

8.6.1 AcitveMQ rozhraní pro spouštění business procesu

V této kapitole změníme způsob spouštění business procesu. Nahradíme současné řešení, které očekává vložení správně formátovaného souboru do požadované složky rozhraním Apache ActiveMQ. V našem příkladu vytvoříme následující Camel routu:

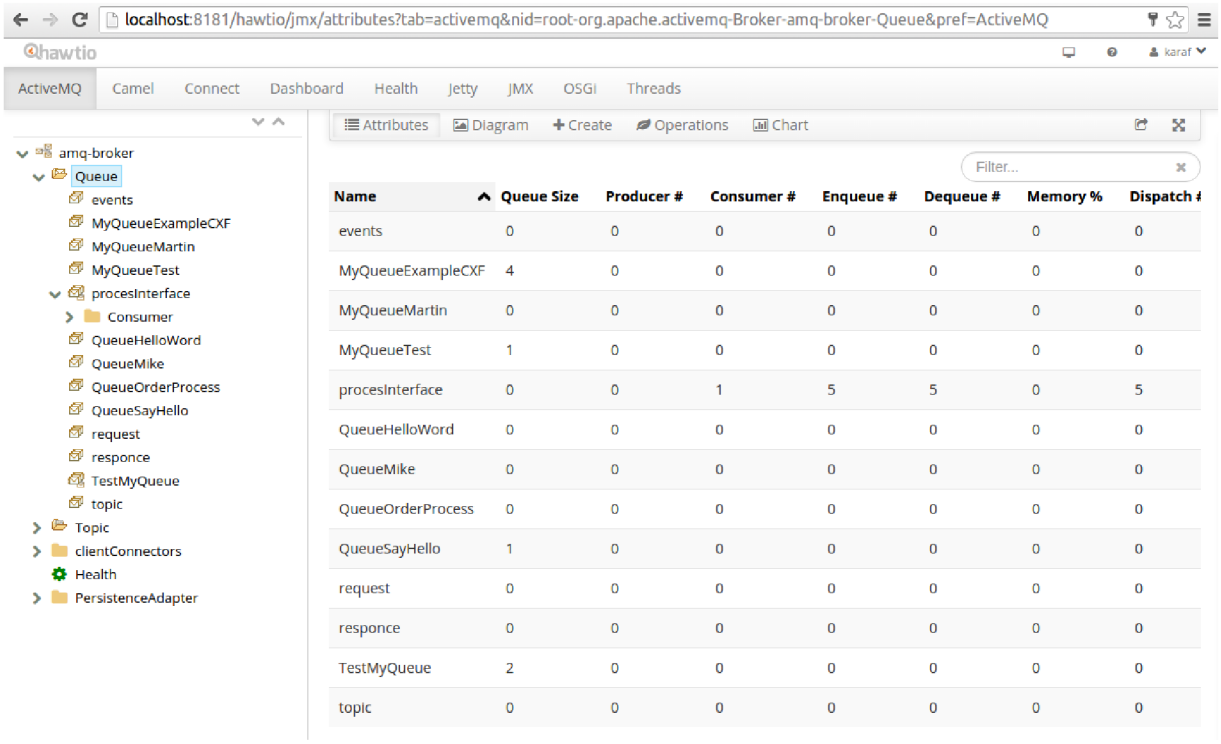
```
from("activemq:queue:procesInterface")
    .setBody(bean(procesInstance))
    .setProperty(PROCESS_KEY_PROPERTY,
                simple(java.util.UUID.randomUUID().toString()))
    .to("activiti:OrderProcess");
```

Touto routou říkáme, že proces spustíme po přijetí zprávy na AcitveMQ rozhraní `procesInterface`. Oproti předchozímu příkladu jsme pozměnili vytváření id procesu. Předtím jsme jako primární klíč používali název souboru. Toto řešení mohlo způsobovat problémy, protože mohly vzniknout dvě aktivní instance procesu se stejným názvem. Nyní pro vytvoření id procesu použijeme standartní Java UUID generátor.

Vyzkoušet, že se business proces spouští z po příchodu zprávy na `processInterface` rozhraní, můžeme snadno s využitím Hawtio konzole. Jedná se o webovou aplikaci – dashbard, která umožňuje sledovat a kontrolovat provoz na serveru. Kromě mnoha různých grafů umožňuje sledování aktivních Camel rout a také sledování ActiveMQ rozhraní. Můžeme zde zjistit seznam aktivních rozhraní, kolik mají přiřazených konzumentů a také zprávy zasílat. Hawtio konzoli nainstalujeme do ServiceMixu příkazem:

```
feature:install hawtio-core
```

S Hawtio konzolí bude moci pracovat po zadání uživatelského jména a hesla „karaf“ na adrese: http://localhost:8181/hawtio/.

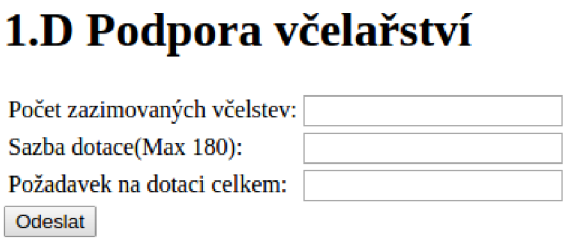


Obrázek 14 - Hawtio konzole

8.6.2 Zasílání zpráv na ActiveMQ rozhraní

Použitím rozhraní ActiveMQ jsme získali širokou škálu možností, jak na toto rozhraní posílat zprávy z klientů vytvořených v různých programovacích jazycích. Zasílání zpráv lze zajistit prostřednictvím protokolu Stomp nebo OpenWire, jejichž implementace je dostupná téměř ve všech dnes používaných programovacích jazycích.

V této kapitole ukážeme, jak s využitím protokolu Stomp vytvořit klienta ve skriptovacím jazyce php. Klient bude webová stránka, která bude umožňovat odeslání formuláře žádosti. Nejprve pomocí html vytvoříme jednoduchý formulář žádosti viz. Obrázek 15 - Formulář žádosti



Obrázek 15 - Formulář žádosti

Pro odeslání zprávy použijeme php skript. Využíváme v něm třídu `Stomp`, které do konstruktoru předáme adresu lokálního ActiveMQ rozhraní a přihlašovací údaje. Defaultní uživatel pro ActiveMQ spuštěné pod Apache ServiceMix je karaf s heslem karaf. Zprávu odesíláme pomocí

metody `send`, které předáváme v parametrech jméno fronty, na kterou posíláme a samotnou zprávu. Důvodům pro zvolené formátování zprávy se budeme věnovat následující kapitole 8.7 Transformace.

Php skript pro odeslání zprávy na AcitveMQ rozhraní

```
$queue  = 'procesInterface';
$msg = '<?xml version="1.0" encoding="UTF-8"?>
<voko:models>
<xf:model>
    <xf:instance xmlns="" id="form-data">
    <instance>
        <PredmetyDotace>
            <PredmetDotace>
                <Kod>1.D.a.</Kod>
                <Sazba>'. $_POST['sazba'] .'</Sazba>
                <SazbaPriznano></SazbaPriznano>
                <MaxSazba></MaxSazba>
                <PocetVcelstev>'.$_POST['pocetVcelstev'].'</PocetVcelstev>
                <PocetVcelstevPriznano></PocetVcelstevPriznano>
                <Pozadovano>'. $_POST['pozadovano'] .'</Pozadovano>
                <Priznano></Priznano>
            </PredmetDotace>
        </PredmetyDotace>
        <ProgramDotace>1.D.</ProgramDotace>
        <PozadovanoCelkem/>
        <PriznanoCelkem/>
    </instance>
    </xf:instance>
</xf:model>
</voko:models>';

$msg = trim_all($msg);
try {
  $stomp = new Stomp('tcp://localhost:61613','karaf','karaf');
  $stomp->send($queue, $msg);
} catch(StompException $e) {
  die('Connection failed: ' . $e->getMessage());
```

Pro fungování toho skriptu je třeba nainstalovat php Stomp a povolit používání v nastavení webového serveru. Dále je třeba do konfiguračního souboru activemq.xml přidat `transportConnector`:

```
<transportConnector name="stomp" uri="stomp://localhost:61613"/>
```

8.7 Transformace

V této kapitole ukážeme, jak Camel routě transformovat zprávy. Pomocí transformací získáváme široké možnosti způsobu napojení jednotlivých komponent systému. Právě možnost využití transformací je jeden z důvodů, proč pro voláme webovou službu z Camel routy namísto z komponenty typu `WebTask` z business procesu.

V našem příkladu ukážeme transformaci na převodu zprávy přijaté z webového formuláře do formátu zprávy pro zavolání webové služby Registr žádostí - Formální kontrola. Bude se tedy jednat o transformaci z xml do xml. Vstupní xml bude respektovat formát modelu kompatibilní s XForms.

Pomocí této technologie jsou vytvořeny všechny formuláře žádostí v současné implementaci ISND. Díky transformaci do formátu potřebného pro zavolání webové služby vzniká možnost využití současné implementace formulářů v TO-BE Architektuře Agribus.

V datovém modelu formuláře se nacházejí pole i pro část formuláře přiznáno. V našem příkladu řešíme část přiznáno s využitím `UserTask` a nadefinovaného nového formuláře pro uživatelskou interakci s využitím Activiti Exploreru. Proto tato pole v našem příkladu zůstanou bez využití.

Vstupního xml – Datový model Formuláře 1.D

```
<?xml version="1.0" encoding="UTF-8"?>
<voko:models>
    <xf:model>
        <xf:instance xmlns="" id="form-data">
          <instance>
                <PredmetyDotace>
                    <PredmetDotace>
                        <Kod>1.D.a.</Kod>
                        <Sazba>180</Sazba>
                        <SazbaPriznano></SazbaPriznano>
                        <MaxSazba></MaxSazba>
                        <PocetVcelstev>6</PocetVcelstev>
                        <PocetVcelstevPriznano></PocetVcelstevPriznano>
                        <Pozadovano>1080</Pozadovano>
                        <Priznano></Priznano>
                    </PredmetDotace>
                </PredmetyDotace>
                <ProgramDotace>1.D.</ProgramDotace>
                <PozadovanoCelkem/>
                <PriznanoCelkem/>
            </instance>
        </xf:instance>
    </xf:model>
</voko:models>
```

Výstupní xml – zpráva volání webové služby Registr žádostí - Formální kontrola

```
<?xml version="1.0" encoding="UTF-8"?>
<ns1:registrZadostiIsValid xmlns:ns1="http://vut.pds/isndNamespace/">
    <dotProgram>1.D</dotProgram>
    <dotPredmet>1.D.a</dotPredmet>
    <Pocet>4</Pocet>
    <Sazba>180</Sazba>
    <Pozadovano>1080</Pozadovano>
</ns1:registrZadostiIsValid>
```

Transformaci bude volat v Camel routě spouštějící business proces. Transformaci zavoláme příkazem:

```
.bean(RequestTransformer.class, "transform(${body})")
```

Třídu `RequestTransformer` s metodou `transform` vytvoříme v samostném souboru. Metoda `transform` bude přijímat argument typu String se vstupním xml a navracet bude String s výstupním xml [6].

Pro parsování xml v této metodě jsme použili `DocumentBuilder`. Jedná se o třídu, která poskytuje rozhraní pro získání DOM (Document Object Model). DOM je objektové API pro práci s XML soubory. DOM nahlíží na XML soubor jako na objektovou, stromovou strukturu. Parsování xml se nám tím velmi zjednoduší.

Pro tvorbu třídy `DocumentBuilder` využijeme Třídu `DocumentBuilderFactory`. K vytvoření instance dokumentu použijeme metodu `parse` Třídy `DocumentBuilder`. Tato metoda je přizpůsobená pro práci se soubory, proto musíme vstupní xml ve formátu String převést na `InputSource`. Toho docílíme pomocí třídy `ByteArreayInputStream`.

```
DocumentBuilderFactory dbFactory = DocumentBuilderFactory.newInstance();
DocumentBuilder dBuilder = dbFactory.newDocumentBuilder();
Document doc = dBuilder.parse(new InputSource(
               new ByteArrayInputStream(transformBody.getBytes("utf-8"))));
doc.getDocumentElement().normalize();
```

Pomocí vytvořené instance dokumentu můžeme přistupovat k jednotlivým položkám modelu pomocí DOM api:

```
String dotProgram = doc.getElementsByTagName("ProgramDotace")
                    .item(0).getTextContent();
```

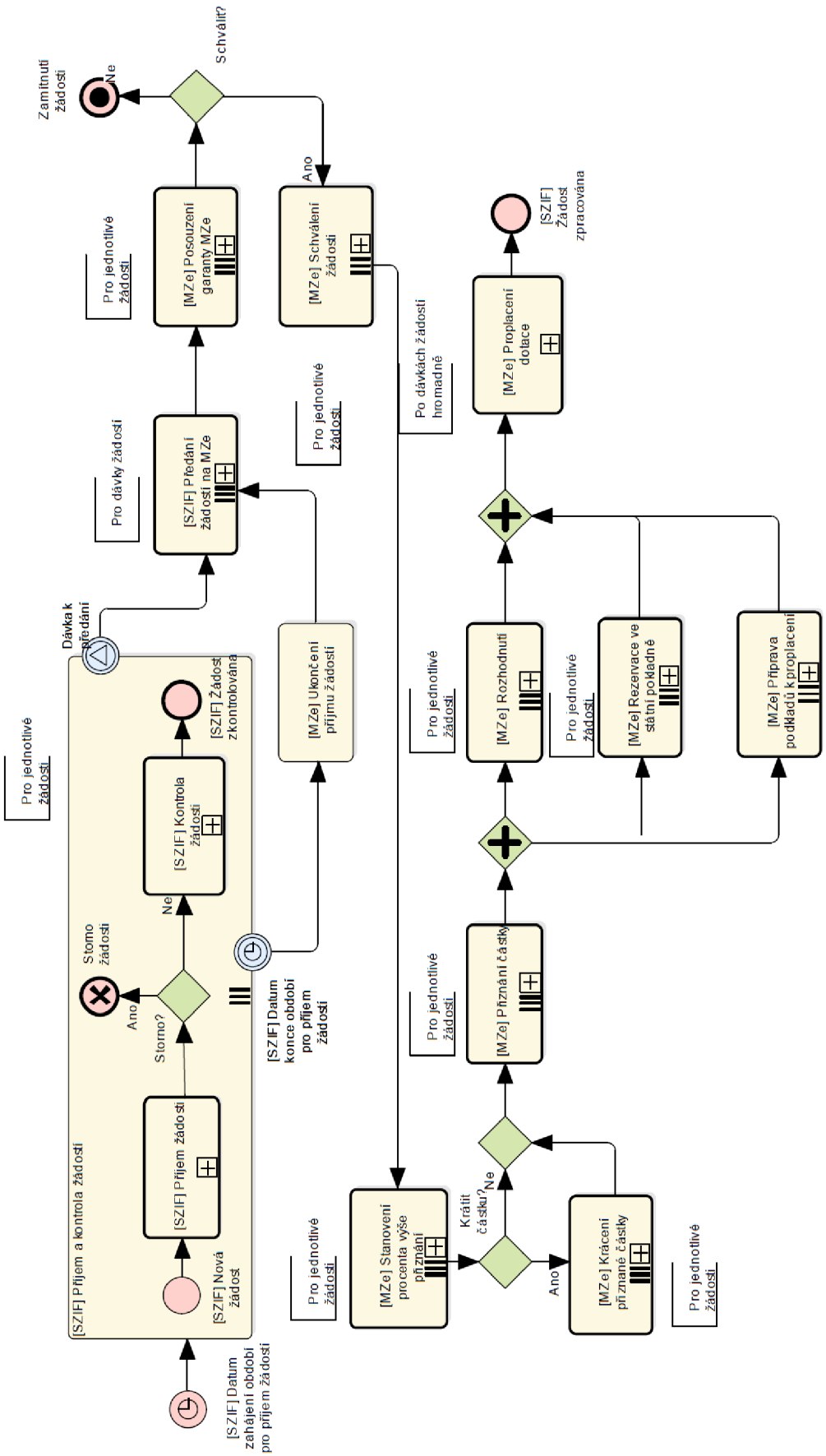
9 Zhodnocení a možnosti rozšíření

V předchozí kapitole jsme ukázali implementaci modelového business procesu – Životní cyklus žádosti. Tento business proces se spouští zprávou poslanou z webového formuláře na ActiveMQ rozhraní s využitím php skriptu a protokolu Stomp. O řízení business procesu se stará Activiti engin instalovaný do Apache ServiceMix. Vytvořený business proces používá komponenty definované pomocí Apache Camelu. Tímto jsme rozšířili možnosti business procesu o možnost routování a transformace zpráv, které Apache Camel umožňuje. V našem modelovém příkladu jsme provedli ukázky transformace zprávy a volání webové služby Registr žádostí – Formální kontrola. Návratovou zprávu z webové služby jsme použili k řízení procesu pomocí brány `exclusiveGateway`.

Dále jsme ukázali uživatelskou interakci s pomocí komponenty procesu typu `UserTask`, která umožňuje definici formuláře. Tuto funkcionalitu jsme prezentovali na ukázce části přiznáno formuláře žádosti o dotace. Aktivní instance procesu s možností vyplnění tohoto formuláře jsme zpřístupnili pomocí Activiti Exploreru.

V případě realizace ESB Agribus by bylo možné na základě ukázaných výsledků rozšířit současný modelový proces životního cyklu žádosti a realizovat úplnou implementaci business procesu zpracování žádosti dle obrázku - Obrázek 16 - Proces zpracování žádosti o dotace. Tento proces využívá nejen komponenty ISND, ale také využívá služeb Státního zemědělského investičního fondu (SZIF). Pro zprovoznění tohoto procesu by bylo nutné všechny tyto služby naimplementovat.

Dále by se dalo pokračovat s rozšiřováním funkcionality ESB AgriBus dle požadavků popsaných v dokumentaci. Jedná se o velice široké spektrum požadavků. Ze začátku by bylo možné implementovat Registr služeb, napojit kompatibilní designer BPM procesů a vytvořit nebo vybrat uživatelsky přívětivý designer webových formulářů.



Obrázek 16 - Proces zpracování žádosti o dotace

Závěr

V rámci této diplomové práce byl představen Informační systém pro národní dotace a jeho budoucí úloha v TO-BE architektuře AgriBus.

Dále jsme představili vizi Ministerstva zemědělství ČR na TO-BE architekturu AgriBus a detailně jsme prozkoumali definované požadavky. Na základě těchto požadavků jsme vytvořili porovnávací kritéria pro výběr ESB, kterým tyto požadavky můžeme naplnit. Na základě těchto kritérií jsme porovnali nejrozšířenější open source ESB řešení a vybrali Apache ServiceMix. Apache ServiceMix jsme detailně porovnali s podrobnými technickými požadavky na ESB Agribus a zjistili, že všechny jsou řešitelné.

V následující kapitole jsme vytvořili plán na nasazení Apache ServiceMix a jeho integraci se systémem národních dotací.

Dále jsme ukázali nutné úpravy současného řešení ISND, aby byl připraven plnit svou plánovanou funkci v TO-BE architektuře AgriBus. Tyto úpravy spočívají v předělání vybraných kontrolerů na webové služby komunikující prostřednictvím protokolu SOAP. Pro jednu z vybraných služeb Registr žádostí – Formální kontrola jsme nadefinovali rozhraní, které jsme implementovali prostřednictvím wsdl. Tuto službu jsme pro další využití simulovali prostřednictvím programu SoapUI.

V následující kapitole jsme ukázali implementaci části ESB Agribus, která bude využívat webové služby ISND. K implementaci jsme využili Apache ServiceMix, Apache Camel, Activiti BPM a Apache ActiveMQ a webový formulář, který odesílá data na ActiveMQ rozhraní pomocí php skriptu. Pro uživatelskou interakci s business procesy jsme použili Activiti Explorer. Možnosti spolupráce těchto prvků jsme demonstrovali na modelovém příkladu business procesu – životní cyklus žádosti o dotace.

V závěrečné kapitole jsme shrnuli komponenty použité k implementaci a ukázali možnosti rozšíření implementovaného příkladu.

V této diplomové práci jsme ukázali, že ESB Agribus lze implementovat s využitím open source řešení Apache ServiceMix, a že ISND bude po mírných úpravách připraven na novou roli v rámci ESB Agribus.

Literatura

[1] Veřejná zakázka: Informační systém Národní dotace 2016. Veřejné zakázky Ministerstva zemědělství [online]. Praha: Ministerstvo zemědělství ČR, 2015 [cit. 2015-10-01]. Dostupné z: https://zakazky.eagri.cz/contract_display_4591.html

[2] Vybudování a provoz komunikační infrastruktury MZe - AgriBus: Zadávací dokumentace příloha č. 4 - Technická specifikace. Veřejné zakázky Ministerstva zemědělství[online]. Praha: Ministerstvo zemědělství ČR, 2014 [cit. 2015-10-01]. Dostupné z: https://zakazky.eagri.cz/contract_display_4591.html

[3] Apache ServiceMix [online]. Apache Software Foundation, 2008 [cit. 2016-01-19]. Dostupné z: http://servicemix.apache.org/

[4] Activiti BPM Platform [online]. [cit. 2016-01-19]. Dostupné z: http://activiti.org/

[5] RADEMAKERS, Tijs. *Activiti in action: executable business processes in BPMN 2.0.* Shelter Island, NY: Manning Publications, c2012. ISBN 1617290122.

[6] IBSEN, Claus. a Jonathan. ANSTEY. *Camel in Action.* Greenwich, Conn.: Manning, c2011. ISBN 1935182366.

Seznam obrázků

Příloha A

Obsah DVD

- /Doc – Technická zpráva.
- /Sources/ProcessExample – Zdrojové kódy modelového procesu životního cyklu žádosti.
- /Sources/ExampleForm.php – Formulář žádosti spouštějící business proces žádosti.
- /Sources/ISND.wsdl – Definice rozhraní webové služby Registr žádostí – Formální kontrola.
- /Sources/ISND-soapui-project.xml – Projektový soubor pro program SoapUI simulující ISND.
- /Sources/README – Návod pro spuštění modelového procesu v přiloženém virtuálním prostředí.
- /Dip-Orava-virtual-Lubuntu.zip – Komprimovaný archiv obsahující virtuální prostředí Lubuntu 14.04 LST (32-bit) ve formátu vdi. Virtuální prostředí lze zprovoznit pomocí aplikace VirtualBox. Pro bezproblémový chod doporučujeme 2 GB paměti.