ОРИГИНАЛ

ПРИЛОЖЕНИЕ № 2б

ТЕХНИЧЕСКО ПРЕДЛОЖЕНИЕ

За участие в открита процедура за възлагане на обществена поръчка с предмет:

„Последващо развитие и усъвършенстване на информационно-комуникационната среда на електронното правителство“*, обособена позиция № 2*

УВАЖАЕМИ ГОСПОЖИ И ГОСПОДА,

След запознаване с документацията за участие в процедурата за възлагане на обществена поръчка за услуга с горе цитирания предмет

Ние, “БУЛ ЕС АЙ” ООД, ЕИК 131423631

адрес на управление :1415 гр.София, кв.Драгалевци, ул.”Христина Морфова” №39, представлявано от Красимир Благоев Антонов-управител,

с Банкова сметка:

IBAN BG52UNCR70001520477529
банков код: UNCRBGSF
банка: УниКредит Булбанк АД

правим следното техническо предложение:

Заявяваме, че ще изпълним поръчката в съответствие с всички нормативни изисквания за този вид дейност, както и в съответствие с изискванията, посочени в техническата спецификация на Възложителя. Декларираме, че сме съгласни с поставените от Вас условия и ги приемаме без възражения.

Предлагаме следния подход за изпълнение:

1 ОРГАНИЗАЦИЯ НА РАБОТАТА

Бул Ес Ай ООД е технологична и консултантска компания, специализирана в разработването на комплексни информационни системи и решения. Компанията разполага с екип висококвалифицирани професионалисти в областта на информационните технологии и има опит в реализацията на комплексни ИТ проекти, притежава експертизата в изграждането на решения в областта на електронното управление, включващи както планиране и изграждане на инфраструктурата и интеграция на базовите софтуерни компоненти, така и реализация на приложния софтуер.

В настоящото техническо предложение след внимателен анализ на изискванията на Възложителя са определени и описани Дейностите, които БУЛ ЕС АЙ ООД, наричана по-долу Изпълнител, ще реализира при изпълнение на ОБОСОБЕНА ПОЗИЦИЯ № 2 от обществена поръчка с предмет: *„Последващо развитие и усъвършенстване на информационно-комуникационната среда на електронното правителство“*. За

ефективната им реализация Изпълнителят ще създаде интердисциплинарен екип от експерти. Експертите отговарят напълно на изискванията в тръжната документация.

По време на реализацията на проекта Изпълнителят ще работи в тясно взаимодействие с Възложителя. Всички резултати, които представляват проучвания, анализ или предложения за промени на и трансформации на бизнес процесите, текущо ще се съгласуват с Възложителя и след окончателното им представяне ще се одобряват от него. Когато следваща задача по дейността зависи от резултатите от предходната задача, работа по нея ще се започва след одобряване на резултатите по предходната.

В това техническо предложение е представена подробна методология за реализация на проекта, включваща подходите при извършване на всяка от дейностите и задачите.

За реализирането на изискванията от настоящето техническо задание предлагаме да се използват комбинирано две световно утвърдени методологии - методологията за цялостно управление на проекти на PMI (Project Management Institute) и методология за използване на единен (унифициран) процес (RUP) за разработването на софтуера. Управлението на дейностите по проекта ще следва методологията на PMI, а за разработката на софтуера ще се следва унифицирания процес RUP за разработка на софтуер.

1.1 Методология за управление на проекта

Въз основа на натрупания опит ние предлагаме използването на методологията на PMI (Project Management Institute) за цялостното управление на проекта. По-долу са описани принципите на тази методология, базирана на *Project Management Body of Knowledge (PMBOK).*

Project Management Body of Knowledge (PMBOK) е сбор от процеси и сфери на знание, широко приети като най-добра практика в дисциплината "Управление на проекти". Този международно признат стандарт (IEEE Std 1490-1998) е основата на управлението на проекти. Според *PMBOK* съществуват 5 основни групи процеси (стартиране, планиране, изпълнение, проследяване и контрол, приключване) и 9 сфери на знание (управление на интеграцията на проекта, на обхвата, на времето, на разходите, на качеството, на човешките ресурси, на комуникациите, на риска и на доставките). Във всеки проект или фаза процесите се застъпват и си взаимодействат. Те се описват от гледна точка на вход (документи, планове, проекти), инструменти и техники (механизми, прилагани върху входящите данни) и изход (документи, продукти, резултати).

Основните цели на методологията на PMI са:

- Контролиране на обхвата, графика, разходите и качеството,
- Намаляване и управляване на риска,
- Управление на ресурсите,
- Идентифициране на дейностите по проекта,
- Координиране на комуникациите между заинтересованите страни,
- Съобразяване на работата с бизнес целите на Възложителя.

За постигане на горните цели методологията е съсредоточена върху следните 9 сфери на знание:

- Управление на интеграцията,
- Управление на обхвата,
- Управление на времето,
- Управление на разходите,
- Управление на качеството,
- Управление на човешките ресурси,
- Управление на комуникациите,
- Управление на риска,
- Управление на доставките.

Процесите по управление на проекта са организирани в пет групи:

- **Стартирането** включва процесите, които се изпълняват при възлагането на роли и определянето на обхвата на нова фаза или проект.
- **Планирането** включва процесите, които се изпълняват при определянето и промяната на обхвата на проекта, разработването на плана за управление на проекта и планирането на дейностите по проекта.
- **Изпълнението** включва процесите по извършване на зададената работа и постигане на целите на проекта, залегнали в обхвата.
- **Проследяването и контролът** включва процесите, необходими за стартирането, планирането, изпълнението и приключването на проекта в съответствие с целите, зададени в обхвата и плана за управление на проекта.
- **Приключването** включва процесите, които се изпълняват при официалното прекратяване на всички дейности по дадена фаза или проект и предаването на готовия продукт.

Всяка група процеси се състои от един или повече управленски процеси. Групите са свързани – често изходът на даден процес се превръща във вход на друг. При централните групи процеси има итерация на връзките — планирането осигурява на изпълнението първоначален документиран план на проекта, след което осигурява актуализации на плана в хода на работата.

Кратко описание на деветте сфери на знание съгласно стандарта на PMI:

- **Управление на интеграцията**

Процесите по управление на интеграцията гарантират правилната координация на различните елементи на проекта. Те включват балансиране на целите и алтернативите с оглед на нуждите и очакванията на заинтересованите страни. Описаните в тази глава процеси са предимно интегративни.

Разработване на план на проекта

При разработването на плана на проекта се използват резултатите от други планиращи процеси, включително стратегическо планиране, за да се създаде един ясен и последователен документ, който да насочва и изпълнението, и контрола на проекта. Този процес минава през няколко итерации. Сборът от всички интегрирани планове за управленски контрол съставлява обхвата на проекта.

Изпълнение на плана на проекта

Изпълнението на плана на проекта е основен процес при осъществяването на плана – преобладаваща част от бюджета и усилията по проекта се изразходват при извършването на този процес. Чрез него ръководителят на проекта и неговия екип координират и насочват техническите и организационните интерфейси. В рамките на този процес фактически се създава продуктът на проекта. Изпълнението постоянно ще се сравнява с основния план на проекта, за да се вземат своевременни коригиращи мерки. В подкрепа на анализа ще се правят периодични прогнози за окончателните разходи и резултати.

Интегриран контрол на промените

Интегрираният контрол на промените се занимава с факторите, които влияят върху пораждането на промени, грижи се за съгласуването на промените, констатира наличието на промени и ги управлява, когато възникнат.

Първоначално дефинираният обхват и интегрираният основен план на проекта се поддържат чрез постоянно управление на възникналите промени чрез приемане или отхвърляне на промените и включването им в актуализираната версия на основния план. Интегрираният контрол на промените изисква:

- Поддържане интегритета на базовите измерители на изпълнението.
- Отразяване на промените в обхвата на продукта във вече дефинирания обхват.
- Координиране на промените във всички сфери на знание.

- **Управление на обхвата**

Управлението на обхвата на проекта включва процесите, които гарантират, че проектът включва цялата необходима работа и само необходимата работа за успешното осъществяване на проекта. То се занимава най-вече с определянето и контролирането на това какво е включено и какво не е включено в проекта.

- Стартирането е процесът на официалното възлагане на нов проект. Официалното възлагане на този проект ще бъде подписването на договор, което ще свърже проекта с работата на изпълнителя.
- Планирането на обхвата е процесът на детайлизиране и документиране на работата по проекта (обхвата на проекта), чийто резултат ще бъде продуктът на проекта. Описанието на продукта обхваща изискванията, които отразяват съгласуваните нужди на клиента, и дизайн, който отговаря на тези изисквания. Резултатите от планирането на обхвата са Дефиниция на обхвата и План за управление на обхвата. Дефиницията на

обхвата е основата за постигане на споразумение между възложителя и изпълнителя, чрез идентифициране на целите и резултатите по проекта. След стартирането на проекта екипите разработват множество дефиниции на обхвата, в съответствие с нивото на детайлизиране на работата (напр. Системен анализ, подробен график и др.).

- Определянето на обхвата включва разбиването на основните резултати, посочени в Дефиницията на обхвата, на по-малки, по-управляеми елементи. Целта е:
  - Подобряване на прогнозите за разходи, продължителност и ресурси.
  - Определяне на основни параметри за измерване на изпълнението и контрол.
  - Ясно разпределяне на отговорностите.
- Потвърждаването на обхвата е процесът по официално приемане на обхвата на проекта от заинтересованите страни. Той изисква преглед на резултатите от работата и потвърждение, че всичко е свършено както трябва. Ако проектът се прекратява преждевременно, потвърждението на обхвата трябва да документира нивото и степента на завършеност.
- Контролът на промените в обхвата се занимава с факторите, които влияят върху пораждането на промени, грижи се за съгласуването на промените, констатира наличието на промени и ги управлява, когато възникнат.

- **Управление на времето**

Управлението на времето по проекта включва следните процеси, необходими за навременното приключване на проекта:

- Определяне на дейностите – идентифициране и документиране на конкретните дейности, необходими за постигане на набелязаните резултатите и под-резултати. Определянето на дейностите се съгласува с Дефиницията на обхвата и включва детайлизиране, предположения и ограничения.
- Последователност на дейностите - идентифициране и документиране на логическите взаимозависимости. Дейностите трябва да бъдат в правилна последователност, за да спомогнат за разработването на реалистичен и постижим график. Последователността може да следва критичната пътека. В резултат се определя график със съответните контролни точки и зависимости.
- Продължителност на дейностите – определя се въз основа на информацията за обхвата на проекта и ресурсите. Предварителната оценка ще се детайлизира в хода на работата, предвид наличието и качеството на входящите данни. Оценката се прави по методологията на критичната пътека.

- Определяне на график – задава се началната и крайната дата на дейностите по проекта. Процесът преминава през няколко итерации преди окончателното определяне на графика на проекта.
- Контрол на графика – занимава се с факторите, които влияят върху пораждането на промени, грижи се за съгласуването на промените, констатира наличието на промени и ги управлява, когато възникнат.

- **Управление на разходите**
  - Планирането на ресурсите включва определяне на количеството и качеството на необходимите ресурси (хора, техника, материали), както и сроковете на тяхното използване. То е тясно свързано с оценката на разходите.
  - Оценката на разходите включва прогнозно определяне на разходите за необходимите ресурси. Взимат се предвид причините за отклонение от окончателната прогноза, за да се осигури по-добро управление на проекта.
  - Бюджетирането на разходите включва разпределяне на общите прогнозни разходи по отделни дейности или групи дейности, за да се установи базовата цена, спрямо която ще се измерва изпълнението. Действителността може да наложи изготвяне на прогнози след одобрението на бюджета, но по възможност те трябва да се правят предварително.
  - Контролът на разходите се занимава с факторите, които влияят върху пораждането на промени, грижи се за съгласуването на промените, констатира наличието на промени и ги управлява, когато възникнат. Контролът на разходите включва:
    - Проследяване изпълнението на бюджета, за да се открият и разберат разминаванията с плана.
    - Точно отразяване на необходимите промени в базовата цена.
    - Предотвратяване на включването на ненужни или неразрешени промени в базовата цена.
    - Информиране на съответната страна за одобрени промени.
    - Осъществяване на очакваните разходи в приемливи граници.

- **Управление на качеството**

Целта на процесите по управление на качеството е да бъдат задоволени нуждите, заради които е предприет проекта. Тези процеси включват всички дейности от цялостното управление на проекта, които определят политиката, целите и отговорностите по качеството и ги осъществяват чрез планиране на качеството, гарантиране на качеството, качествен контрол и подобряване на качеството в рамките на системата за качество.

- *Планиране на качеството* – идентифициране на стандартите за качество за конкретния проект и начините за спазването им. Това е един от ключовите процеси при планиране на качеството и ще се извършва редовно, успоредно с останалите процеси по планиране на проекта.
- *Гарантиране на качеството* – всички планирани и систематични действия в рамките на системата за качество, които дават увереност, че проектът ще отговаря на съответните стандарти. Ще се извърша в хода на целия проект от вътрешни Специалисти по качеството.
- *Качествен контрол* – проследяване на конкретни резултати, за да се определи дали отговарят на зададените стандарти и да се набележат начини за отстраняване на причините за незадоволителните резултати. Ще се извърша в хода на целия проект. Резултатите включват както доставката на конкретен резултат/ продукт, така и резултати от управлението на проекта (изпълнение на бюджета и графика). Би било полезно да се знае разликата между:
  - Предотвратяване (недопускане на грешки в процеса) и проверка (недопускане на грешки от страна на клиента).
  - Изпробване на атрибути (резултатът отговаря или не отговаря) и изпробване на променливи (резултатите се измерват по прогресивна скала за степен на съответствие).
  - Специални причини (необичайни събития) и случайни причини (нормално отклонение от процеса).
  - Допустимост (резултатът е приемлив, ако попада в посочения обхват на допустимост) и контролни граници (процесът е под контрол, ако резултатът е в рамките на контролните граници).

- **Управление на човешките ресурси**

Управлението на човешките ресурси включва процесите, които осигуряват най-ефективното използване на хората, участващи в проекта. То обхваща всички заинтересовани страни – клиенти, партньори, индивидуални изпълнители и др. Състои се от:

- Организационно планиране — идентифициране, документиране и определяне на роли, отговорности и канали за отчитане.
- Набиране на персонал — осигуряване на необходимите човешки ресурси и включването им в работата по проекта.
- Развитие на екипа — развиване на индивидуални и групови умения, с цел подобряване на изпълнението.

- **Управление на комуникациите**

Процесите по управление на комуникациите осигуряват навременното и адекватно генериране, събиране, разпространение, съхранение и унищожаване на информацията по проекта. Те осъществяват критичната за успеха връзка между хора, идеи и данни.

Всеки участник в проекта трябва да е готов да изпраща и приема комуникации и трябва да разбира как каналът на комуникация, в която участва, се отразява на целия проект.

- *Планиране на комуникациите* – определяне на нуждите на заинтересованите страни от информация и комуникации: кой от каква информация се нуждае, как ще я получи и от кого. Нуждата от предоставяне на информация за проекта е общовалидна, но информационните нужди и методите на разпространение са различни за всеки проект. Идентифицирането на нуждата от информация и разпространяването й по подходящ начин е важен фактор за успех на проекта.
- *Разпространение на информацията* – своевременното достигане на информацията до заинтересованите страни. Включва прилагането на Плана за комуникация и откликването на неочаквани искания на информация.
- *Отчитане на изпълнението* – събиране и разпространение на данни за изпълнението, показателни за използването на ресурсите за постигане на целите на проекта. Този процес включва:
  - Отчитане на състоянието — описва докъде е стигнал проектът в дадения момент,
  - Отчитане на напредъка — описва какво е постигнал екипът по проекта,
  - Прогнозиране — предполага бъдещото състояние и напредък по проекта.
  - Отчитане на изпълнението – данни за обхвата, графика, разходите и качеството.
- *Административно приключване*: след постигане на целите или след прекратяване по други причини, проектът или фазата трябва да приключи. Административното приключване се състои от документиране на резултатите, с цел официалното приемане на продукта от страна на клиента. То включва събиране на проектната документация, която отразява окончателните спецификации, анализ на успеха и ефективността на проекта и на извлечените поуки, и архивиране на тази информация за бъдещо ползване. Дейностите по административното приключване не се отлагат до приключването на проекта. Всяка фаза трябва да бъде надлежно приключена, за да не бъде загубена тази важна и полезна информация.

- **Управление на риска**

Управлението на риска е систематичният процес по идентифициране, анализиране и реагиране на рисковете по проекта. То включва максимализиране на вероятността и последствията от благоприятни събития и минимизиране на вероятността и последствията от нежелателни за проекта събития. Проектният риск е несигурно

събитие или състояние, което, ако се случи, има положително или отрицателно влияние върху целите на проекта. Рискът има причина и, ако се материализира, последствие.

- *Планиране на управлението на риска* - процесът на определяне на подхода и дейностите по управление на риска. Важно е да се планират и последващите процеси по управление на риска, за да има съизмеримост между нивото, вида и прозрачността на управление на риска от една страна и самия и риск и важността на проекта за организацията от друга.
- *Идентификация на риска* – определяне на рисковете, които могат да повлияят на проекта, и документирането на техните характеристики. Участници в процеса на определяне на риска са: екипът по проекта, екипът по управление на риска, специалисти от други клонове на фирмата, клиенти, крайни потребители, други ръководители на проекти и външни експерти. Определянето на риска е итеративен процес. Първата итерация може да се осъществи от част от екипа по проекта или от екипа по управление на риска. Целият екип по проекта и основните заинтересовани лица могат да осъществят втората итерация. Щом бъде идентифициран даден риск, се разработват и дори внедряват прости и ефективни мерки за преодоляването му.
- *Качествен анализ на риска* – оценка на влиянието и вероятността от даден риск. Този процес приоритизира рисковете според евентуалното им влияние върху целите на проекта. Качественият анализ на риска е един от начините за определяне важността на дадени рискове и насочване на усилията към справяне с тях. Времето за реакция може да е критичен фактор при някои рискове. Оценката на качеството на наличната информация също спомага при преоценката на риска. Качественият анализ на риска изисква оценка на вероятностите и последствията, чрез установени методи и инструменти.
- *Количественият анализ* на риска е цифровото изражение на вероятността от даден риск и последствията му върху целите на проекта. В този процес ще се използва техника, базирана на опростяване на симулацията "Монте Карло" и анализ на решенията, с цел:
  - Определяне на вероятността за постигане на дадена цел по проекта.
  - Изчисляване на вероятностите за излагане на проекта на риск и определяне на резервни разходи и график.
  - Откриване на рисковете, които изискват най-голямо внимание, чрез изчисляване на относителната им тежест за проекта.
  - Идентифициране на реалистични и постижими разходи, график или обхват.
- *Планирането на реакции на риска* е процесът на разработване на варианти и определяне на действия, които увеличават възможностите и намаляват заплахите за осъществяване целите на проекта. Той включва възлагане на отговорности на отделни лица или групи във връзка с

действията при отделните рискове. Този процес гарантира адекватна реакция на идентифицираните рискове. Ефективността на планирането на реакции е пряко свързана с увеличаването или намаляването на рисковете по проекта.

- *Наблюдението и контролът на риска* е процесът по проследяване на идентифицираните рискове, наблюдаване на остатъчни рискове и отриване на нови рискове. Той спомага за осъществяването на плановете за риска и оценката на ефективността им. Това е постоянен процес в хода на проекта. С времето рисковете се променят, появяват се нови, някои очаквани рискове не се материализират. Доброто наблюдение и контрол на рисковете дава информация, която подпомага взимането на ефективни решения преди материализирането на риска.

Контролът на риска може да включва избор на алтернативна стратегия, прибягване до резервен план, извършване на коригиращи действия или повторно планиране на проекта. Ръководителят на проекта и ръководителят на екипа за риска периодично получават информация на ефективността на плана и наличието на неочаквани влияния и взимат съответните мерки в хода на проекта.

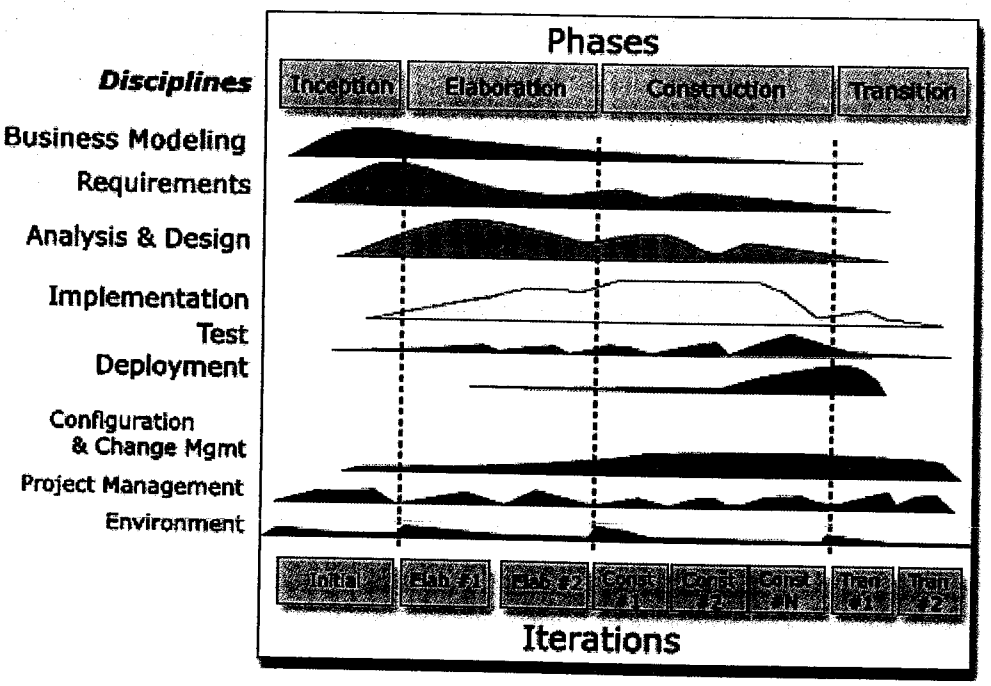
- **Управление на доставките**

Управлението на доставките от трети лица се занимава с придобиването на стоки и услуги от външни за изпълнителя организации. Този процес се състои от:

- Планиране на доставките
- Планиране на търсенето
- Търсене
- Избор на източник
- Администриране на договори
- Приключване на договори

1.2 Методология за разработка на софтуер

Предлаганата методология е базирана на единния рационален процес за разработка на софтуер (RUP), която се използва от водещи ИТ фирми. Единният процес е процес на итеративно разработване на софтуер - адаптивна рамка, която описва начините за ефективно разработване на софтуер с помощта на доказани техники. Той обхваща голям брой дейности, но дава възможност за подбор на най-подходящите от тях за осъществяването на конкретния проект. Единният процес е особено приложим при големи екипи от разработчици, ангажирани с мащабни проекти. Процесът е описан на цикли, фази, итерации и основни етапи. С прилагането на тази методология ще се осигури управлението на всички фази и дисциплини, свързани с жизнения цикъл на текущата софтуерна разработка.



Фигура - Жизнен цикъл, дисциплини и фази

Дисциплини

RUP предлага девет дисциплини за управление, както е посочено на фигурата по-горе.

1. *Дисциплина „Бизнес моделиране"*

Дисциплината „Бизнес моделиране" осигурява използване на общ език за комуникация в бизнес модела и в софтуерния модел чрез описание на бизнес процесите с бизнес use cases. Тази дисциплина е опционална за всяка конкретна софтуерна разработка, тъй като е насочена към оптимизация на бизнес процесите в организацията на Възложителя и не винаги е свързана с последваща софтуерна разработка.

2. *Дисциплина „Изисквания"*

Дисциплината „Изисквания" дефинира най-общо какво трябва да прави системата и подпомага заинтересованите страни в софтуерния проект да постигнат съгласие относно обхвата и границите на разработката.

3. *Дисциплина „Анализ и дизайн"*

Дисциплината „Анализ и дизайн" е изключително важна, защото описва как системата ще се реализира при имплементирането. Тази дисциплина е основополагаща за архитектурния модел на системата, който обхваща различни гледни точки за модела.

4. *Дисциплина „Имплементиране"*

Дисциплината „Имплементиране" позволява системата да се изгради чрез създаване на модули и компоненти, които ще се интегрират в изпълними файлове.

5. *Дисциплина „Тестване*

Дисциплината „Тестване" позволява да се открият колкото е възможно по-рано евентуалните г и проблемни места, с което да се редуцират средства и усилията за отстраняването им.

6. *Дисциплина „Внедряване"*

Дисциплината „Внедряване" осигурява успешното изпълнение на софтуерните версии и доставката им до крайните потребители.

7. *Дисциплина „Управление на проекта"*

Дисциплината „Управление на проекта" осигурява прилагането на итеративен подход и успешно балансиране между факторите, влияещи на изпълнението като цяло.

8. *Дисциплина „Обкръжаваща среда"*

Дисциплината „Обкръжаваща среда" подсигурява процеса на разработка като предоставя методологии, стратегии, процедури, правила и ръководства за работа, шаблони, програмни средства и др.

9. *Дисциплината „Конфигуриране и управление на промените"*

Дисциплината „Конфигуриране и управление на промените" осигурява единен подход за номериране/идентифициране на артефактите, създавани от различни хора, в рамките на проекта. Също така осигурява управлението на исканията за промяна.

Жизнен цикъл

Жизнените цикли на софтуерния продукт се разбиват на индивидуални цикли на разработка, които от своя стана се разбиват на отделни компоненти, наричани фази, а именно Планиране, Проектиране, Изграждане и Предаване.

RUP има четири основни етапа, които съответстват на четирите фази. Ако критериите за тези основни етапи не са изпълнени, проектът може да бъде спрян или да се направи нова итерация. Този мета-модел подчертава връзката между приключването на фазите, итерациите и основните етапи.

Единният процес изисква добра комуникация между изпълнителите на проекта и тясно сътрудничество с бенефициента. Методологията включва визуални техники за моделиране, които използват Единен език за моделиране (UML).

Фаза „Планиране"

Главната цел на фаза „Планиране" е определянето на насоките на проекта и създаване на организация за изпълнението на проектите цели, отнесени към жизнения цикъл на софтуерната разработка.

Основните дейности за фазата са:

- Определяне на обхвата на проекта на концептуално ниво;
- Синтезиране на архитектурата относно технологии и съществуващи компоненти;

- Планиране и подготовка на работата по проекта, в това число обхват, проектен екип, итерации по фази, инфраструктура, рискове и други планове, според спецификата на проекта.

Основни артефакти от фазата:

- Визия за проекта с дефиниран концептуален обхват на разработката;
- Модел на изискванията (концептуален модел за обхвата на системата);
- План за разработване на системата (План на проекта) - съдържа цялата информация, необходима за организиране на проекта, списък с идентифицираните рисковете, график на проекта и други специфични планове (например, План за комуникация, План за управление на промените). План за разработване на системата ще може да се актуализира при необходимост и да се предава за одобрение, като резултат в следващите фази.
- План за управление на качеството - дава рамката, процедурите и правилата за осигуряване качеството. Съдържа информация за планирани прегледи, стратегия за тестване, образци на проектите документи по RUP и др.
- Оценка за изпълнението на фазата - Междинен доклад за напредъка по проекта

Фаза „Проектиране"

Целта на фаза „Детайлизиране" е да се постигне стабилност на архитектурата, плановете и изискванията и да се минимизират техническите рискове, влияещи на успеха на следващите фази.

Основните дейности за фазата са:

- Проектиране на визията (обобщаване и детайлизиране на изискванията);
- Проектиране на процеса и инфраструктурата;
- Проектиране на архитектурата и компонентите й. Архитектурното решение се верифицира от Възложителя в предоставен дизайн на модулите, обектен модел и движението на информацията между обектите съобразени с избраната технология от Изпълнителя;

Основни артефакти от фазата са:

- Модел на изискванията (допълнен и подобрен модел за обхвата на системата): Документи за Детайлната софтуерна спецификация и Спецификацията на допълнителните изисквания, съдържащи диаграми и текстови описания на изискванията;
- Архитектурен модел: Документ, съдържащ описание на ключовите работни сценарии, които влияят на архитектурата, дизайн на модулите, обектен модел и движението на информацията между обектите, процеси за изграждане и внедряване на системата. Съдържа още описание на архитектура позволяваща скалируемост и осигуряваща непрекъснат режим на работа (7x24), уеб-услугите, приложните сървъри, базите данни. Документът описва предложените технически решения.
- Технически модел: Документ, съдържащ диаграми и описание на елементите, от които се състои системата (обекти, класове, интерфейси, връзки и т.н.). В

описанието се представя йерархията, взаимовръзките, взаимодействията, интерфейсите. Техническият модел се разработва итеративно.

- Тестов модел: Набор от Тестови сценарии, базирани на работните сценарии на системата, съгласно Модела на изискванията, разработен в същата фаза.

Фаза „Разработка"

Целта на тази фаза е разработването на завършен софтуерен продукт, който да се подготви за предаване на потребителите. Усилията са съсредоточени върху разработката на софтуера (писане на програмен код и реализиране на вътрешни тестове).

Основните дейности за фазата са:

- Управление на ресурсите, контролиране и оптимизиране на процеса поизграждане на системата;
- Завършване разработката на компоненти/модули и тестването им;
- Оценяване на версиите на системата спрямо изискванията. В тази фаза се идентифицират и реализират нови/пропуснати изисквания.

Основни артефакти от фазата са:

- Използваеми версии на системата: Системата се разработва постепенно, като във всяка нова версия се добавя нова функционалност от специфицираните работни сценарии.
- План на проекта - актуализирана версия;
- Тестов план: Документ за организиране на тестовете по приемане;
- План за внедряване: Съдържа описание на дейностите и ресурсите (хардуер, софтуер, персонал), необходими за инсталирането и тестването на разработения продукт, с цел ефективното му предаване на потребителите;
- План за обучение на потребители: Съдържа описание на дейностите и ресурсите, необходими за провеждане на обучението, в това число - програма на обучението и график за провеждането му.
- Оценка за изпълнението на фазата - Междинен доклад за напредъка по проекта;

Фаза „Предаване/въвеждане в експлоатация"

Целта на тази фаза е да се предостави реализираната система на крайните й потребители. Пълната версия на системата се внедрява и се интегрира в съществуващата инфраструктура на Възложителя. Други цели на фазата са провеждането на тестове за приемане и обучение на крайните потребители.

Основните дейности за фазата са:

- Тестове по приемане на реализирания продукт и оценяването им. Тестовете се провеждат в присъствието на Изпълнителя в съответствие с Тестовите сценарии и Тестовия план, предадени във фаза „Изграждане";
- Фина настройка на продукта на база събраната информация от клиента по време на тестовете, отстраняване на бъгове, повишаване на производителността при необходимост;
- Обучение на потребители за работа със системата;

- Внедряване на системата за експлоатация;
- Предаване на реализацията и съпровождащата й потребителска и техническа документация.
- Организиране на гаранционната поддръжка.

Основни артефакти от тази фаза са:

- Завършена система (окончателна версия): Завършени и тествани модули на системата;
- Оценка на резултатите от тестовете на системата: Документ, който обобщава резултатите от тестовете по приемане и съдържа тестова статистика;
- Ръководство за потребителите и техническа документация за администраторите на системата: Потребителска документация се изготвя за всяка конкретна потребителска група.
- Учебни материали: документация, подпомагаща провеждането на обучение, организирано за потребителите на системата.
- Изходен код: предава се заедно с окончателната версия на системата (в зависимост от договорните взаимоотношения).
- План за поддръжка: Описва обхвата на поддръжката и процедурата, която се ще се осигури гаранционната поддръжка.
- Оценка за изпълнението на проекта - Окончателен доклад за изпълнението на проекта;

Всички фази и итерации на проекта включват дейности по управление на проекта, управление на конфигурацията, управление на промяната и управление на средата (инструменти , стандарти за кодиране, документация, хранилища, системи за контрол на версиите).

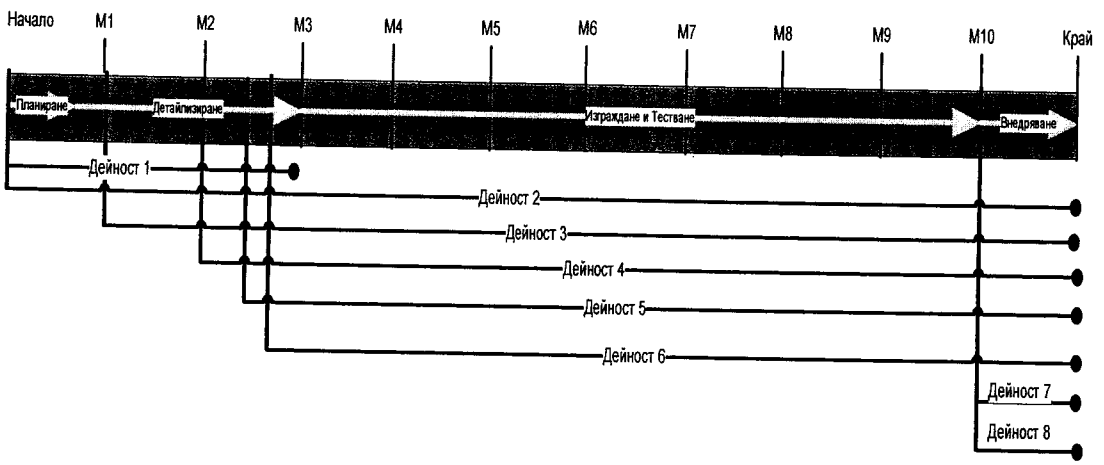
2 ДЕЙНОСТИ И ФАЗИ

Представения график за изпълнение на проекта е индикативен и следва да се обнови съвместно с Възложителя в рамките на 7 дни след подписване на договора. Съгласно техническото задание срокът за изпълнение на проекта е 11 месеца, считано от датата на подписване на договора. Гаранционната поддръжка на разработените модули е с продължителност 36 месеца, считано от датата на датата на приемане на системите в експлоатация.

Графикът съдържа основните дейности от обхвата на проекта, позиционирани по фази във времето за изпълнение, основните зависимости между отделните фази и момента на доставка на отчетните продукти от проекта.

В хода на проекта, при възникване на необходимост, графикът ще бъде актуализиран по съгласувана Процедура за управление на промените, част от съгласувания унифициран процес, който е част от Плана за управление на проекта.

Следващата фигура показва обобщената времева линия за изпълнение на дейностите от обособена позиция 2.



Обобщен времеви график

В таблицата са представени резултатите, които са съобразени с изискванията от Техническо задание за етапи на изпълнение и тяхната последователност, пречупени през жизнения цикъл и итеративния подход на софтуерната разработка, съгласно предложената методология за управление (RUP).

Фаза	**Ключов резултат**	**Роля**
Общи	Месечни доклади Окончателен доклад	Ръководител на проекта

Планиране	План за управление на проекта План за управление на качеството Списък с рискове Речник План за следваща фаза	Ръководител на проекта Бизнес аналитик Системен архитект
Детайлизиране	Анализ на текущото състояние Функционална и техническа спецификация за Дейност 2, 3, 4, 5, 6 Системен проскт за Дейност 2, 3, 4, 5, 6	Бизнес аналитик Системен архитект Р-л софтуерна разработка
Изграждане	Изходен код и инсталационен пакет Актуализирана функционална и техническа спецификация за Дейност 2, 3, 4, 5, 6	Р-л софтуерна разработка Програмисти
Тестване	План за тестване Тестови сценарии Резултати от тестове Ръководства за работа План за внедряване	Тестери Специалист по качество Бизнес аналитик
Предаване	Отчет от внедряване Отчет от обучение	Системен архитект Р-л софтуерна разработка

В предложения индикативен график за изпълнение е видно, че броя итерации за отделните фази е различен за всяка една от дейностите. По-голяма част от дейностите ще се изпълняват със застъпване, тъй като имат зависимости по между си.

В рамките на Дейност 1 ще се извърши обективен анализ на разработените до момента концепция на БеУ, софтуерна архитектура на БеУ, електронна идентификация на БеУ, както и съществуващите и разработвани в момента проекти за нормативна уредба на БеУ. Изпълнението на дейността планираме да се изпълни по време на фазите Планиране и Детайлизиране.

В рамките на Дейност 2 ще се разработи компонент за централизирана еднократна автентикация на заявители на електронни услуги, служители в администрацията, както и на информационни системи пред системите на БеУ, като делта на автентикацията е

да бъде установена валидността между крайния потребител и неговата претендирана самоличност. Изпълнението на тази дейност ще обхване всички фази, като фазите Планиране и Предаване ще имат по една итерация, Детайлизиране – две, а Изграждане – три.

В рамките на Дейност 3 ще се разработи компонент за електронна оторизация, позволяващ дефиниране на гъвкави правила за разрешаване или ограничаване на достъпа до системни ресурси. Изпълнението на тази дейност ще обхване всички фази, като фазите Планиране и Предаване ще имат по една итерация, Детайлизиране – две, а Изграждане – три.

В рамките на Дейност 4 ще се доразвие шината за услуги (ESB) за връзка с компонентите за еднократна автентикация и електронна оторизация. Изпълнението на тази дейност ще обхване всички фази, като фазите Планиране и Предаване ще имат по една итерация, Детайлизиране – една, а Изграждане – две.

В рамките на Дейност 5 ще се разработи система за генериране и обработка на бизнес събития. Изпълнението на тази дейност ще обхване всички фази, като като фазите Планиране, Детайлизиране и Предаване ще имат по една итерация, а Изграждане – две.

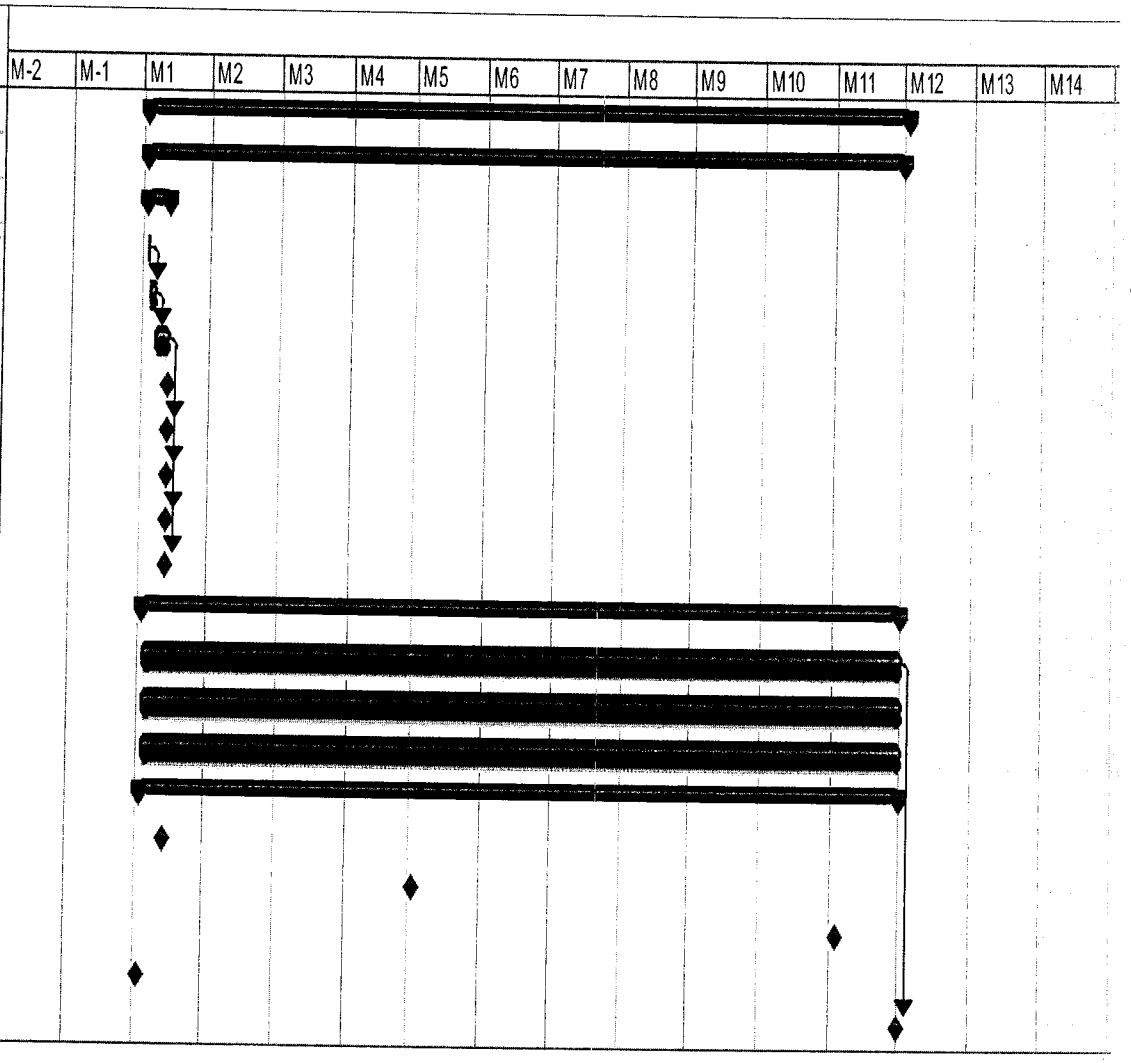
В рамките на Дейност 6 ще се разработи журнал на достъпа до ресурси в БеУ, основаващ се на системата за генериране и обработка на бизнес събития. Изпълнението на тази дейност ще обхване всички фази, като фазите Планиране, Детайлизиране и Предаване ще имат по една итерация, а Изграждане – две.

В рамките на Дейност 7 ще се извърши интеграция между решенията, разработени по дейности 2, 3, 4, 5 и 6. Изпълнението на тази дейност ще се проведе изцяло във фаза Внедряване.

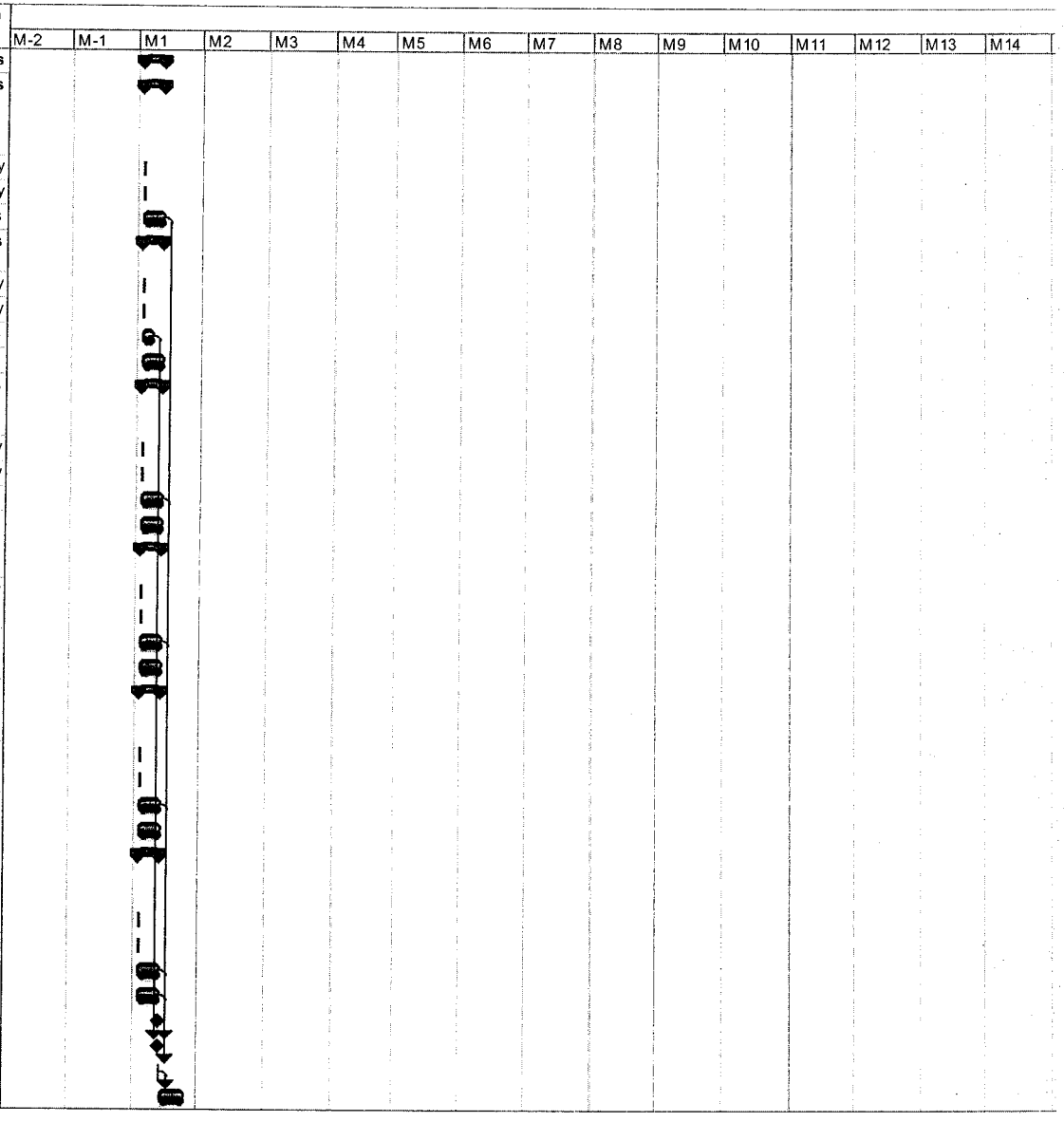
В рамките на Дейност 8 ще се извърши обучение за работа с разработените продукти. Изпълнението на тази дейност ще се проведе изцяло във фаза Внедряване.

Индикативен график за изпълнение

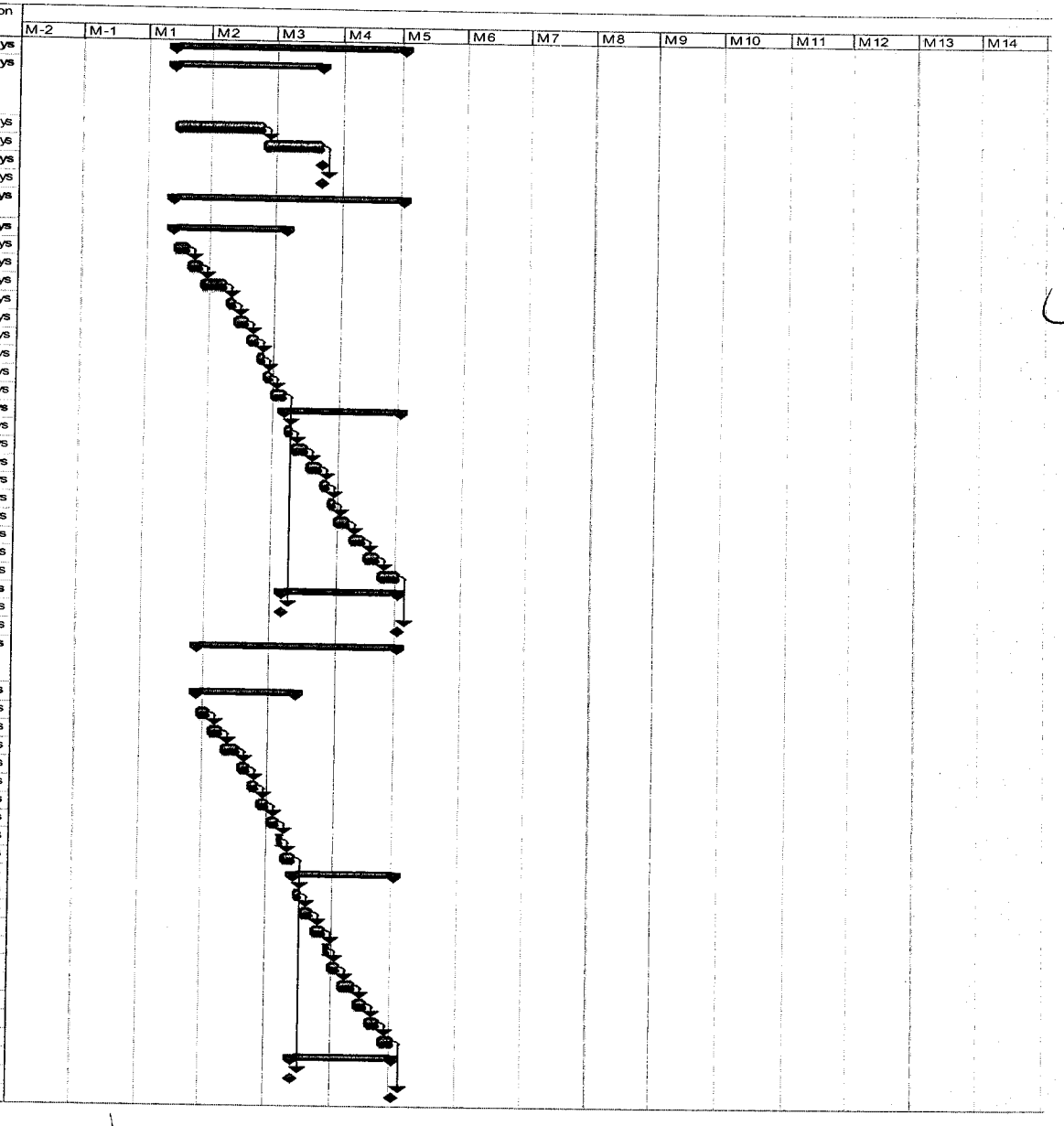
ID	Task Name	Duration
1	МТИТС	39,7 days
2	Управление на проекта	238 days
3	Стартиране на проекта	8 days
4	Среща за стартиране на проекта	1 day
5	Сформиране на екип и определяне на ресурсите от Възл	3 days
6	Изготвяне на отчетни резултати	4 days
7	Резултати	0 days
8	План за управление на проекта	0 days
9	План за управление на качеството	0 days
10	Списък с рискове	0 days
11	Детайлен план за фаза Планиране	0 days
12	Общи	238 days
13	Управление на проекта	238 days
14	Управление на качеството	238 days
15	Управление на риска	238 days
16	Резултати	238 days
17	Детайлен план за фаза Детайлизиране	0 days
18	Детайлен план за фаза Изграждане	0 days
19	Детайлен план за фаза Внедряване	0 days
20	Месечни доклади	0 days
21	Окончателен доклад	0 days



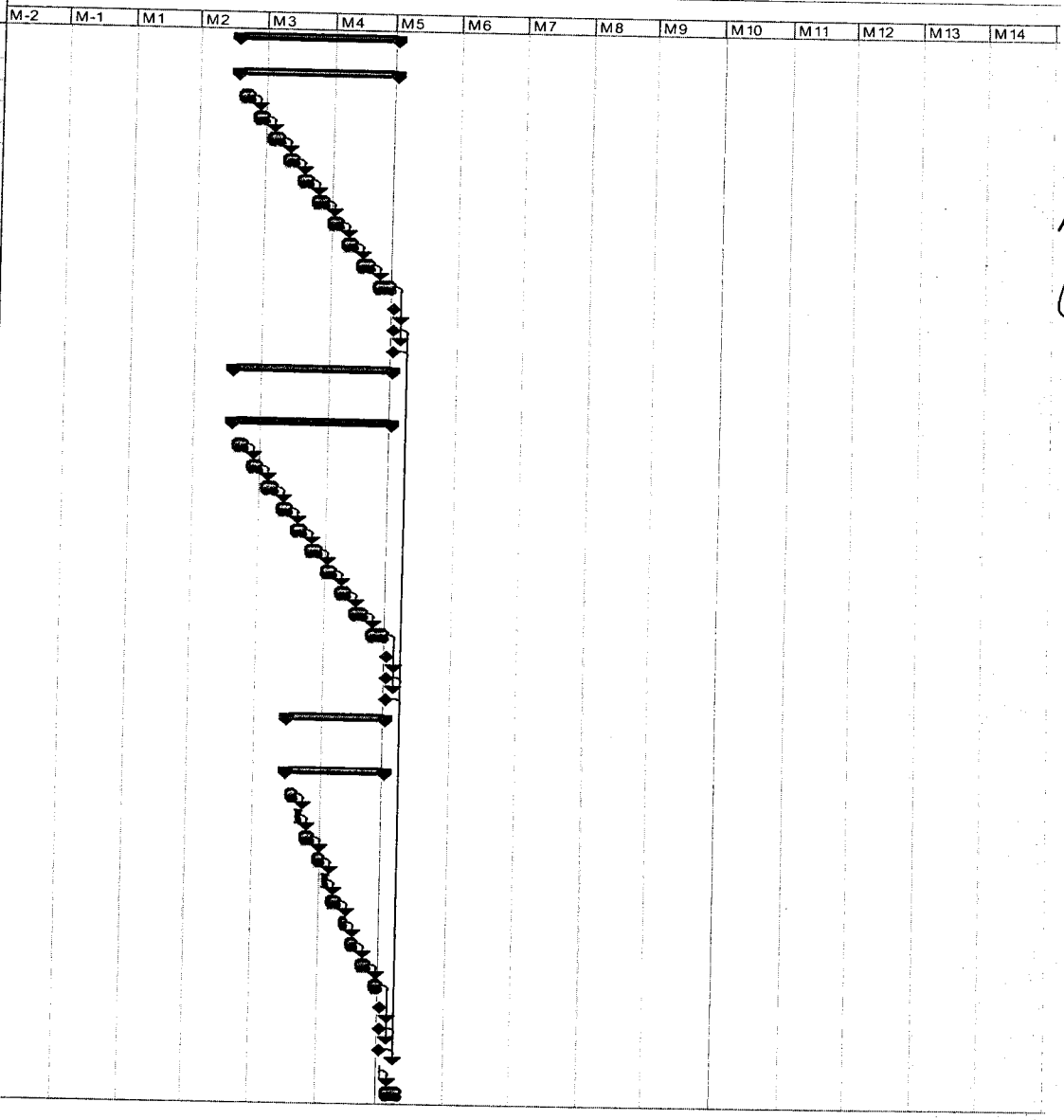
ID	Task Name	Duration
22	**Планиране**	**8 days**
23	**Дейност 1: Анализ на разработените до момента концепция на БеУ, софтуерна архитектура на БеУ, електронна идентификация на БеУ, както и съществуващите и разработвани в момента проекти**	**8 days**
24	Сформиране на екип	1 day
25	Изготвяне на план за работа	1 day
26	Стартиране на анализа на съществуващото в момента	8 days
27	**Дейност 2: Разработване на компонент за еднократна автентикация на ИС и/или служители в**	**8 days**
28	Сформиране на екип	1 day
29	Изготвяне на план за работа	1 day
30	Анализ на функционални и технически изискванията	5 days
31	Подготовка на развойна среда	8 days
32	**Дейност 3: Внедряване на компонент за електронна оторизация, позволяващдефиниране на гъвкави правила за разрешаване или ограничаване на**	**8 days**
33	Сформиране на екип	1 day
34	Изготвяне на план за работа	1 day
35	Анализ на функционални и технически изискванията	8 days
36	Подготовка на развойна среда	8 days
37	**Дейност 4: Развитие на шината за услуги (ESB) за връзка с компонентите за еднократна автентикация и**	**8 days**
38	Сформиране на екип	1 day
39	Изготвяне на план за работа	1 day
40	Анализ на функционални и технически изискванията	8 days
41	Подготовка на развойна среда	8 days
42	**Дейност 5: Разработване на система за генериране и обработка на бизнес събития. Предоставяне на възможност за интеграция на системата с други**	**8 days**
43	Сформиране на екип	1 day
44	Изготвяне на план за работа	1 day
45	Анализ на функционални и технически изискванията	8 days
46	Подготовка на развойна среда	8 days
47	**Дейност 6: Разработване на журнал на достъпа до ресурси в БеУ, основаващсе на системата за генериране и обработка на бизнес събития**	**8 days**
48	Сформиране на екип	1 day
49	Изготвяне на план за работа	1 day
50	Анализ на функционални и технически изискванията	8 days
51	Подготовка на развойна среда	8 days
52	**Резултати**	**0 days**
53	Речник	0 days
54	Предаване на Фаза Планиране	0,5 days
55	Приемане на фаза Планиране	7 days



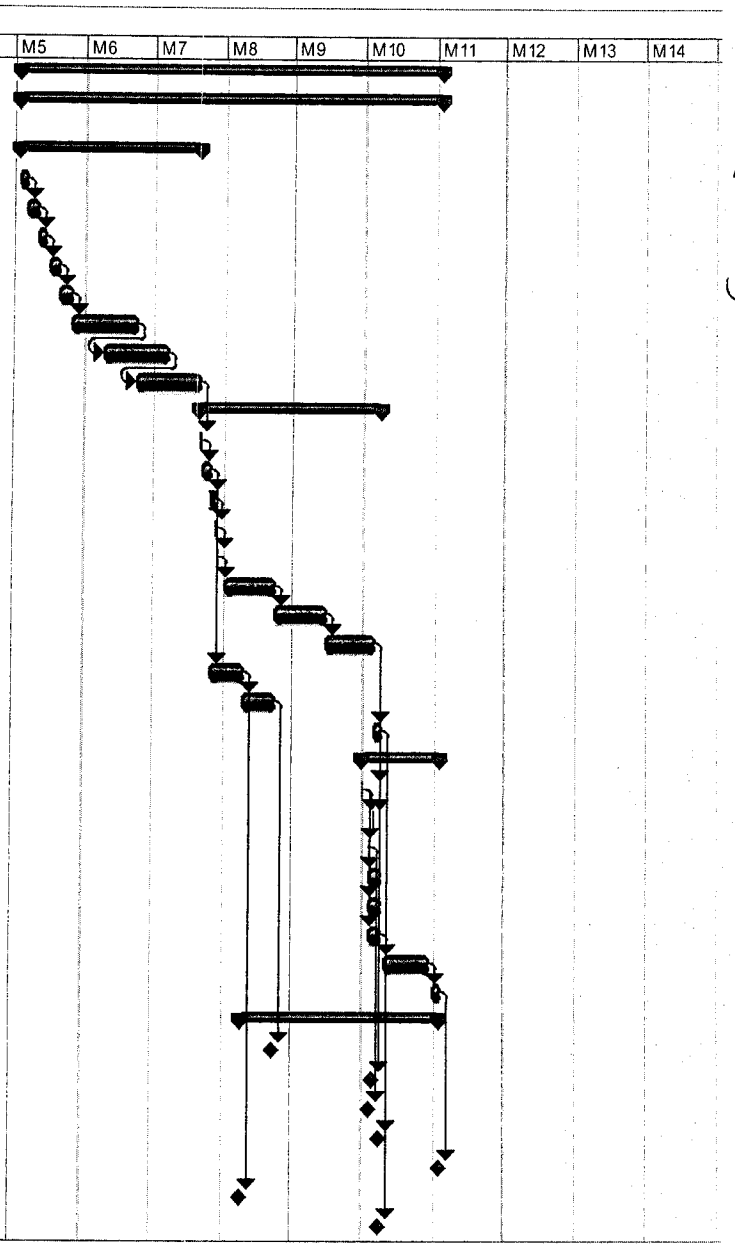
ID	Task Name	Duration
56	**Детайлизиране**	**77 days**
57	**Дейност 1: Анализ на разработените до момента концепция на БеУ, софтуерна архитектура на БеУ, електронна идентификация на БеУ, както и съществуващите и разработвани в момента проекти**	**50 days**
58	Анализ на разработените до момента концепция, архит	30 days
59	Изготвяне на доклад с резултатите от анализа	20 days
60	**Резултати**	**0 days**
61	Доклад с анализ на съществуващото в момента ре	0 days
62	**Дейност 2: Разработване на компонент за еднократна автентикация на ИС и/или служители в**	**77 days**
63	**Итерация 1**	**38 days**
64	Прецизиране на бизнес изискванията	5 days
65	Дефиниране на модел и описание на работни проце	4 days
66	Детайлизиране на модел и описание на случаите на	8 days
67	Дефиниране на потребителски интерфейс	4 days
68	Разработване на прототип и потребителски интерф	4 days
69	Дефиниране на логически и физически модел на д	3 days
70	Дизайн на техническа архитектура на решението	3 days
71	Дизайн на модела данни	2 days
72	Изготвяне на функционална и техническа специфи	5 days
73	**Итерация 2**	**39 days**
74	Финализиране на бизнес изискванията	3 days
75	Финализиране на модел и описание на работни про	5 days
76	Финализиране на модел и описание на случаите на	5 days
77	Финализиране на потребителски интерфейс	2 days
78	Финализиране на логически и физически модел на	3 days
79	Финализиране на дизайн на техническа архитектура	5 days
80	Финализиране на дизайн на модела данни	5 days
81	Изготвяне на софтуерна архитектура	5 days
82	Изготвяне на документ Системен проект	6 days
83	**Резултати**	**39 days**
84	Функционална и техническа спецификация	0 days
85	Системен проект	0 days
86	**Дейност 3: Внедряване на компонент за електронна оторизация, позволяващ дефиниране на гъвкави правила за разрешаване или ограничаване на**	**67 days**
87	**Итерация 1**	**33 days**
88	Прецизиране на бизнес изискванията	4 days
89	Дефиниране на модел и описание на работни проце	4 days
90	Детайлизиране на модел и описание на случаите на	6 days
91	Дефиниране на потребителски интерфейс	3 days
92	Разработване на прототип и потребителски интерф	4 days
93	Дефиниране на логически и физически модел на д	3 days
94	Дизайн на техническа архитектура на решението	3 days
95	Дизайн на модела данни	2 days
96	Изготвяне на функционална и техническа специфи	4 days
97	**Итерация 2**	**34 days**
98	Финализиране на бизнес изискванията	3 days
99	Финализиране на модел и описание на работни про	4 days
100	Финализиране на модел и описание на случаите на	4 days
101	Финализиране на потребителски интерфейс	2 days
102	Финализиране на логически и физически модел на	3 days
103	Финализиране на дизайн на техническа архитектура	5 days
104	Финализиране на дизайн на модела данни	4 days
105	Изготвяне на софтуерна архитектура	4 days
106	Изготвяне на документ Системен проект	5 days
107	**Резултати**	**34 days**
108	Функционална и техническа спецификация	0 days
109	Системен проект	0 days



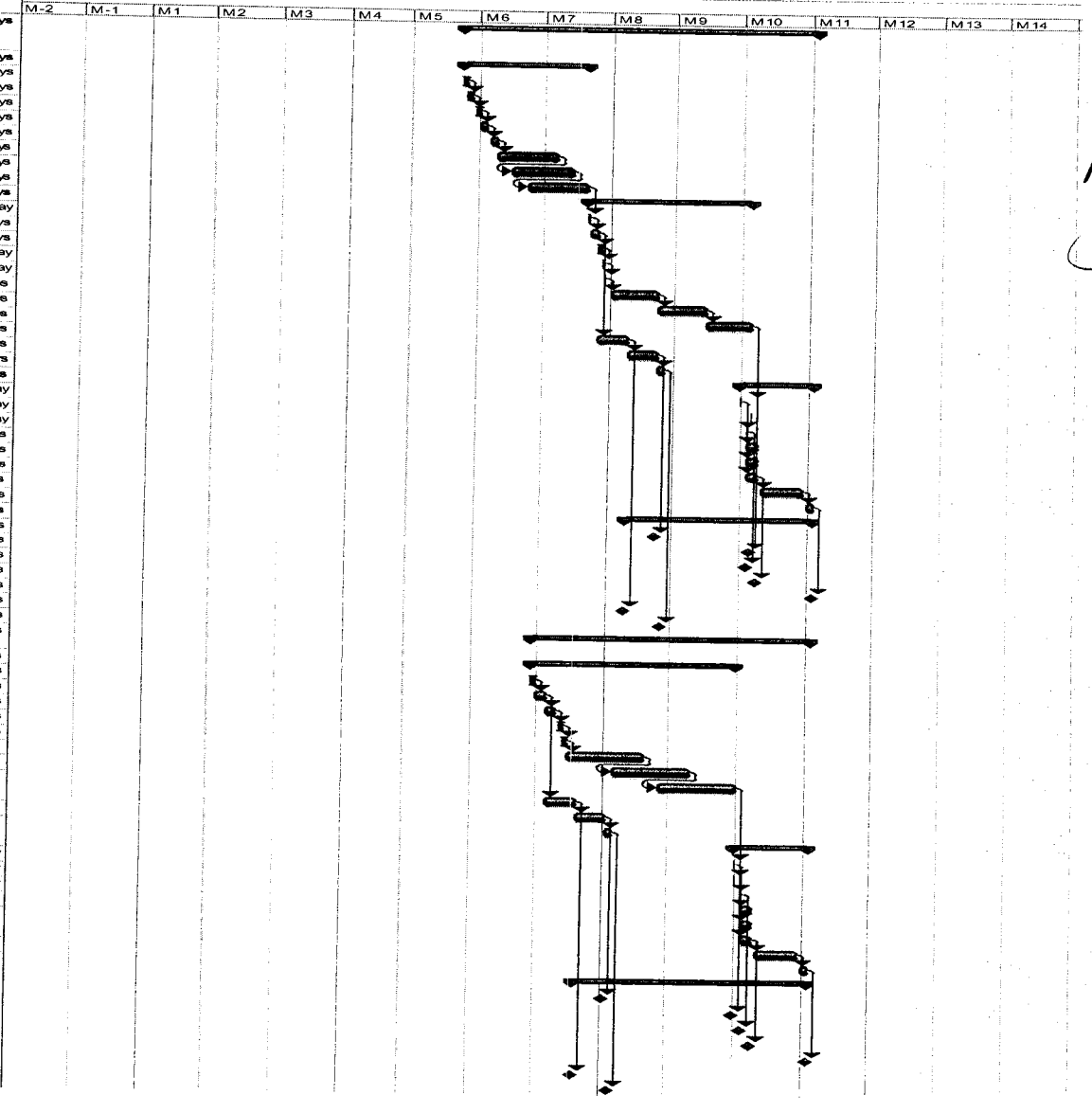
ID	Task Name	Duration
110	**Дейност 4: Развитие на шината за услуги (ESB) за връзка с компонентите за еднократна автентикация и**	**52 days**
111	**Итерация 1**	**52 days**
112	Дефиниране на бизнес изискванията	5 days
113	Дефиниране на модел и описание на работни проце	5 days
114	Дефиниране на модел и описание на случаите на уп	5 days
115	Дефиниране на потребителски интерфейс	5 days
116	Дефиниране на логически и физически модел на д	5 days
117	Дефиниране на дизайн на техническа архитектура н	5 days
118	Дефиниране на дизайн на модела данни	5 days
119	Изготвяне на софтуерна архитектура	5 days
120	Изготвяне на функционална и техническа специфи	6 days
121	Изготвяне на документ Системен проект	6 days
122	**Резултати**	**0 days**
123	Функционална и техническа спецификация	0 days
124	Системен проект	0 days
125	**Дейност 5: Разработване на система за генериране и обработка на бизнес събития. Предоставяне на възможност за интеграция на системата с други**	**52 days**
126	**Итерация 1**	**52 days**
127	Дефиниране на бизнес изискванията	5 days
128	Дефиниране на модел и описание на работни проце	5 days
129	Дефиниране на модел и описание на случаите на уп	5 days
130	Дефиниране на потребителски интерфейс	5 days
131	Дефиниране на логически и физически модел на д	5 days
132	Дефиниране на дизайн на техническа архитектура н	5 days
133	Дефиниране на дизайн на модела данни	5 days
134	Изготвяне на софтуерна архитектура	5 days
135	Изготвяне на функционална и техническа специфи	6 days
136	Изготвяне на документ Системен проект	6 days
137	**Резултати**	**0 days**
138	Функционална и техническа спецификация	0 days
139	Системен проект	0 days
140	**Дейност 6: Разработване на журнал на достъпа до ресурси в БеУ, основаващсе на системата за генериране и обработка на бизнес събития**	**32 days**
141	**Итерация 1**	**32 days**
142	Дефиниране на бизнес изискванията	3 days
143	Дефиниране на модел и описание на работни проце	2 days
144	Дефиниране на модел и описание на случаите на уп	4 days
145	Дефиниране на потребителски интерфейс	3 days
146	Дефиниране на логически и физически модел на д	2 days
147	Дефиниране на дизайн на техническа архитектура н	4 days
148	Дефиниране на дизайн на модела данни	3 days
149	Изготвяне на софтуерна архитектура	3 days
150	Изготвяне на функционална и техническа специфи	4 days
151	Изготвяне на документ Системен проект	4 days
152	**Резултати**	**0 days**
153	Функционална и техническа спецификация	0 days
154	Системен проект	0 days
155	Предаване на Фаза Детайлизиране	0,5 days
156	Приемане на Фаза Детайлизиране	7 days



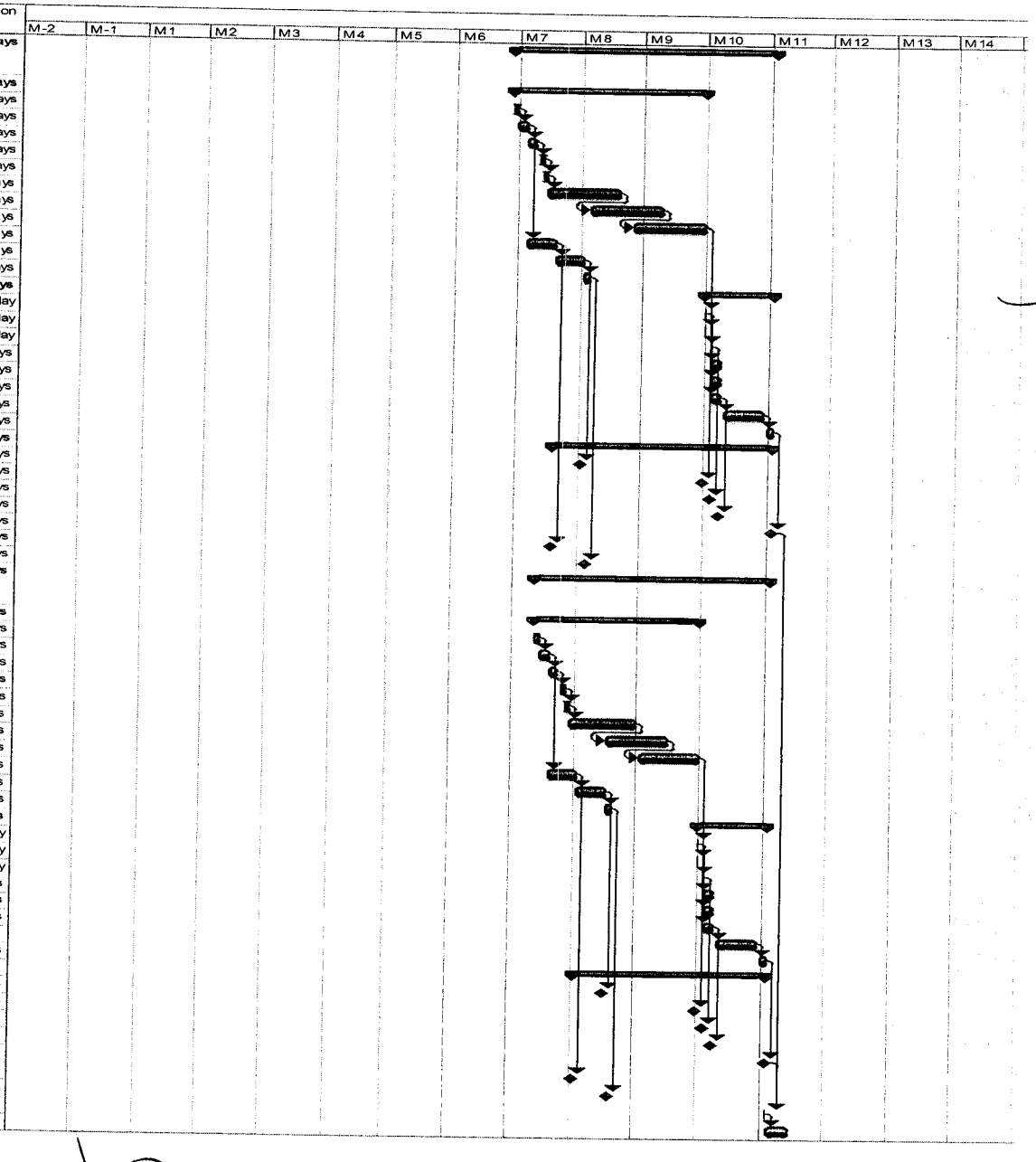
ID	Task Name	Duration
157	**Изграждане**	**132 days**
158	**Дейност 2: Разработване на компонент за еднократна автентикация на ИС и/или служители в**	**132 days**
159	**Итерация 1**	**57 days**
160	Преглед и прецизиране на промените към изисква	3 days
161	Финализиране на дизайн на случаи на употреба	3 days
162	Финализиране дизайн класове	3 days
163	Финализиране дизайн база данни	4 days
164	Прецизиране на архитектура	4 days
165	Разработка на софтуерните модули и компоненти	20 days
166	Unit тестване	20 days
167	Отстраняване дефекти	20 days
168	**Итерация 2**	**57 days**
169	Финализиране на промените към изискванията	1 day
170	Финализиране на дизайн на случаи на употреба	2 days
171	Финализиране дизайн класове	2 days
172	Финализиране дизайн база данни	1 day
173	Финализиране на архитектура	1 day
174	Разработка на софтуерните модули и компоненти	16 days
175	Unit тестване	16 days
176	Отстраняване дефекти	15 days
177	Изготвяне на ръководства за работа	10 days
178	Актуализиране на функционална и техническа спец	10 days
179	Изготвяне на план за внедряване	3 days
180	**Итерация 3 - Тестване**	**24 days**
181	Подготовка на тестова среда	1 day
182	Инсталация и настройка на разработеното решени	1 day
183	Изготвяне на план за тестване	1 day
184	Разработка на тестови сценарии и тестови данни -	5 days
185	Разработка тестови сценарии и тестови данни- инт	5 days
186	Разработка тестови сценарии и тестови данни- про	5 days
187	Изпълнение на тестови сценарии	15 days
188	Обобщаване на резултати от приемателните тест	3 days
189	**Резултати**	**62 days**
190	Актуализиране функционална и техническа специф	0 days
191	Изходен код и инсталационен пакет	0 days
192	План за тестване	0 days
193	Тестови сценарии	0 days
194	Резултати от тестване	0 days
195	Ръководства за работа	0 days
196	План за внедряване	0 days



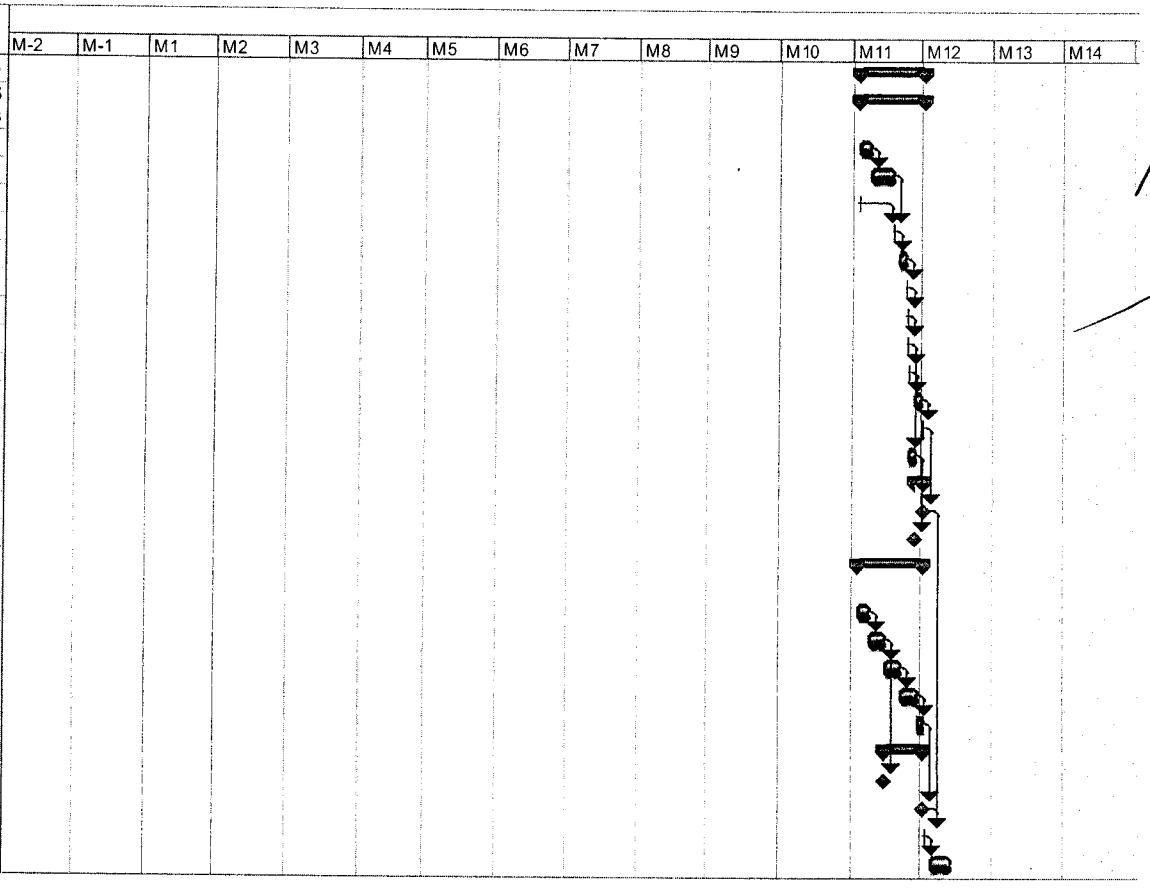
ID	Task Name	Duration
197	**Дейност 3: Внедряване на компонентза електронна оторизация, позволяващдефиниране на гъвкави правила за разрешаване или ограничаване на**	**117 days**
198	**Итерация 1**	**42 days**
199	Преглед и прецизиране на промените към изиска	2 days
200	Финализиране на дизайн на случаи на употреба	2 days
201	Финализиране дизайн класове	2 days
202	Финализиране дизайн база данни	3 days
203	Прецизиране на архитектура	3 days
204	Разработка на софтуерните модули и компоненти	20 days
205	Unit тестване	20 days
206	Отстраняване дефекти	20 days
207	**Итерация 2**	**54 days**
208	Финализиране на промените към изискванията	1 day
209	Финализиране на дизайн на случаи на употреба	2 days
210	Финализиране дизайн класове	2 days
211	Финализиране дизайн база данни	1 day
212	Финализиране на архитектура	1 day
213	Разработка на софтуерните модули и компоненти	16 days
214	Unit тестване	16 days
215	Отстраняване дефекти	15 days
216	Изготвяне на ръководства за работа	10 days
217	Актуализиране на функционална и техническа спец	10 days
218	Изготвяне на план за внедряване	3 days
219	**Итерация 3 - Тестване**	**24 days**
220	Подготовка на тестова среда	1 day
221	Инсталация и настройка на разработеното решени	1 day
222	Изготвяне на план за тестване	1 day
223	Разработка на тестови сценарии и тестови данни -	5 days
224	Разработка тестови сценарии и тестови данни- инт	5 days
225	Разработка тестови сценарии и тестови данни- прс	5 days
226	Изпълнение на тестови сценарии	15 days
227	Обобщаване на резултати от приемателните тестс	3 days
228	**Резултати**	**62 days**
229	Актуализиране функционална и техническа специф	0 days
230	Изходен код и инсталационен пакет	0 days
231	План за тестване	0 days
232	Тестови сценарии	0 days
233	Резултати от тестване	0 days
234	Ръководства за работа	0 days
235	План за внедряване	0 days
236	**Дейност 4: Развитие на шината за услуги (ESB) за връзка с компонентите за еднократна автентикация и**	**92 days**
237	**Итерация 1**	**68 days**
238	Финализиране на промените към изискванията	2 days
239	Финализиране на дизайн на случаи на употреба	3 days
240	Финализиране дизайн класове	4 days
241	Финализиране дизайн база данни	2 days
242	Финализиране на архитектура	2 days
243	Разработка на софтуерните модули и компоненти	25 days
244	Unit тестване	25 days
245	Отстраняване дефекти	25 days
246	Изготвяне на ръководства за работа	10 days
247	Актуализиране на функционална и техническа спец	10 days
248	Изготвяне на план за внедряване	3 days
249	**Итерация 2 - Тестване**	**24 days**
250	Подготовка на тестова среда	1 day
251	Инсталация и настройка на разработеното решени	1 day
252	Изготвяне на план за тестване	1 day
253	Разработка на тестови сценарии и тестови данни -	5 days
254	Разработка тестови сценарии и тестови данни- инт	5 days
255	Разработка тестови сценарии и тестови данни- прс	5 days
256	Изпълнение на тестови сценарии	15 days
257	Обобщаване на резултати от приемателните тестс	3 days
258	**Резултати**	**77 days**
259	Актуализиране функционална и техническа специф	0 days
260	Изходен код и инсталационен пакет	0 days
261	План за тестване	0 days
262	Тестови сценарии	0 days
263	Резултати от тестване	0 days
264	Ръководства за работа	0 days
265	План за внедряване	0 days



ID	Task Name	Duration
266	**Дейност 5: Разработване на система за генериране и обработка на бизнес събития. Предоставяне на възможност за интеграция на системата с други**	**92 days**
267	**Итерация 1**	**68 days**
268	Финализиране на промените към изискванията	2 days
269	Финализиране на дизайн на случаи на употреба	3 days
270	Финализиране дизайн класове	4 days
271	Финализиране дизайн база данни	2 days
272	Финализиране на архитектура	2 days
273	Разработка на софтуерните модули и компоненти	25 days
274	Unit тестване	25 days
275	Отстраняване дефекти	25 days
276	Изготвяне на ръководства за работа	10 days
277	Актуализиране на функционална и техническа спец	10 days
278	Изготвяне на план за внедряване	3 days
279	**Итерация 2 - Тестване**	**24 days**
280	Подготовка на тестова среда	1 day
281	Инсталация и настройка на разработеното решени	1 day
282	Изготвяне на план за тестване	1 day
283	Разработка на тестови сценарии и тестови данни -	5 days
284	Разработка тестови сценарии и тестови данни- инт	5 days
285	Разработка тестови сценарии и тестови данни- прс	5 days
286	Изпълнение на тестови сценарии	15 days
287	Обобщаване на резултати от приемателните тестс	3 days
288	**Резултати**	**77 days**
289	Актуализиране функционална и техническа специф	0 days
290	Изходен код и инсталационен пакет	0 days
291	План за тестване	0 days
292	Тестови сценарии	0 days
293	Резултати от тестване	0 days
294	Ръководства за работа	0 days
295	План за внедряване	0 days
296	**Дейност 6: Разработване на журнал на достъпа до ресурси в БеУ, основаващсе на системата за генериране и обработка на бизнес събития**	**82 days**
297	**Итерация 1**	**58 days**
298	Финализиране на промените към изискванията	2 days
299	Финализиране на дизайн на случаи на употреба	3 days
300	Финализиране дизайн класове	4 days
301	Финализиране дизайн база данни	2 days
302	Финализиране на архитектура	2 days
303	Разработка на софтуерните модули и компоненти	22 days
304	Unit тестване	22 days
305	Отстраняване дефекти	21 days
306	Изготвяне на ръководства за работа	10 days
307	Актуализиране на функционална и техническа спец	10 days
308	Изготвяне на план за внедряване	3 days
309	**Итерация 2 - Тестване**	**24 days**
310	Подготовка на тестова среда	1 day
311	Инсталация и настройка на разработеното решени	1 day
312	Изготвяне на план за тестване	1 day
313	Разработка на тестови сценарии и тестови данни -	5 days
314	Разработка тестови сценарии и тестови данни- инт	5 days
315	Разработка тестови сценарии и тестови данни- прс	5 days
316	Изпълнение на тестови сценарии	15 days
317	Обобщаване на резултати от приемателните тестс	3 days
318	**Резултати**	**67 days**
319	Актуализиране функционална и техническа специф	0 days
320	Изходен код и инсталационен пакет	0 days
321	План за тестване	0 days
322	Тестови сценарии	0 days
323	Резултати от тестване	0 days
324	Ръководства за работа	0 days
325	План за внедряване	0 days
326	Предаване на Фаза Изграждане	0,5 days
327	Приемане на Фаза Изграждане	7 days



ID	Task Name	Duration
328	**Внедряване**	**21 days**
329	**Дейност 7: Интеграция на разработените компоненти и системи в средата на БеУ**	**20,5 days**
330	Планиране на интеграцията между отделните дейности	3 days
331	Интегриране на разработените системи по отделните д	8 days
332	Подготовка на среда за тестване на интеграция	0,5 days
333	Инсталация на разработените системи по отделните д	0,5 days
334	Изпълнение на тестови сценарии в частта интеграция	3 days
335	Подготовка продукционна среда	0,5 days
336	Настройка инструменти и проверка на настройки§инстал	0,5 days
337	Подготовка на инсталационни пакети	0,5 days
338	Изготвяне на release notes	0,5 days
339	Инсталация на комплексното решение	3 days
340	Изготвяне на отчет от внедряването	1 day
341	Изготвяне на процедура за гаранционно поддържане	1 day
342	**Резултати**	**3,5 days**
343	Отчет от внедряване	0 days
344	Процедура за гаранционно поддържане	0 days
345	**Дейност 8: Обучение на 10 обучителя от администрацията за работа с разработените продукти**	**21 days**
346	Подготовка на план за провеждане на обучение по всичк	3 days
347	Съгласуване на плана	5 days
348	Разработване на учебни материали	5 days
349	Провеждане на обучение на 10 човека	5 days
350	Изготвяне на отчет от проведеното обучение	3 days
351	**Резултати**	**13 days**
352	План за провеждане на обучение	0 days
353	Отчет от проведеното обучение	0 days
354	Предаване на Фаза Внедряване	0,5 days
355	Приемане на Фаза Внедряване	7 days



3 РИСКОВЕ И ПРЕДПОСТАВКИ

Като се основаваме на нашия опит при подобни проекти идентифицирахме следните допускания, предпоставки и рискове, които могат да окажат влияние върху изпълнението на проекта. По отношение на управлението на рисковете ще се базираме изцяло на PMI методологията, като подробно описание на използваната методика за управление на рисковете е дадена в точка 5.

Предпоставки и допускания

За оптималното изпълнение на проекта очакваме Възложителя да:

- Да състави проектен екип с представители на възложителя по отделните дейности, които ще ползват резултатите от проекта.
- Осигурява поисканата от Изпълнителя информация в установения срок и с необходимото качество;
- Осигури необходимите специалисти по отделните дейности по време на изпълнение на задачите, свързани с анализ и проектиране;
- Осигури необходимите специалисти за участие в работни срещи по отделните дейности, според предварително уточнен график;
- Осигури необходимите тестови данни

Рискове

Рисковете се приоритизират според това доколко проекта е изложен на съответния риск. За всеки от списъка с рискове е извършена експертна оценка на следните показатели:

- **Приоритет** – получава се като произведение от степента на влияние и вероятността от настъпване;
- **Степен на влияние** – отклонението от планираните графици, усилия и разходи, ако рискът действително се реализира. Степен на теглата – ниска (1), несъществена (2), умерена (3), съществена (4), висока (5);
- **Вероятност от настъпване** – това е вероятността рискът действително да се реализира. Степен на теглата - рискът е елиминиран (0%), минимална (1 - 20%), ниска (21 - 40%), средна (41 - 60%), голяма (61 - 80%), много голяма (81 - 100%);
- **Индикатор** - за всеки риск от списъка се идентифицира подлежащо на измерване състояние, настъпването на което означава, че рискът действително се е реализирал. Тези състояния представляват индикатори (лакмуси) за появата на риска (с чиято помощ разбираме, че рискът е вече реалност);
- **Стратегия за смекчаване -** разработването на планове за ограничаване на риска означава да се намалят последствията от настъпването на риска. За някои рискове се изисква да се опишат и действията, чието изпълнение зависи от настъпването на риска.

Предварителен списък с рискове

№	Описание	Въздействие върху проекта	Собственик (Отговорник	Приоритет	Степен на влияние	Вероятност	Индикатор	Стратегия за смекчаване
1	Промяна на обхвата	Увеличена нужда от ресурси. Изоставане от графика на проекта	Възложител	15	Висока	Средна (41 - 60%)	Брой промени	Потребителските изисквания следва да бъдат документирани много ясно и да бъдат поставени под контрол при искане за промяна.
2	Неясен обхват или разминаване в разбиранията за обхвата между Възложителя и Изпълнителя	Некоректно дефиниран краен резултат и възможно забавяне на предаване му	Възложител/ Изпълнител	15	Висока	Средна (41 - 60%)	Изоставане от графика на проекта	Още в началото обхвата на проекта трябва да бъде ясно дефиниран съгласуван между Възложителя и Изпълнителя, както трябва да бъде съгласувана и ясна процедура за въвеждане и одобряване на промени в обхвата и приоритетите.
3	Възникване на затруднения при изготвяне на техническа спецификация поради неясно или твърде тясно формулирани изисквания в задание	Некоректно дефиниран краен резултат и възможно забавяне на предаване му	Възложител/ Изпълнител	15	Висока	Средна (41 - 60%)	Изоставане от графика на проекта	Още в началото обхвата на проекта трябва да бъде ясно дефиниран, съгласуван между Възложителя и Изпълнителя, както трябва да бъде съгласувана и ясна процедура за въвеждане и одобряване на промени в обхвата и приоритетите.
4	Затруднения при съгласуване на техническа спецификация, свързани с постигане на необходими параметри, използвани технологии, лицензни условия и др	Некоректно дефиниран краен резултат и възможно забавяне на предаване му	Възложител/ Изпълнител	15	Висока	Средна (41 - 60%)	Изоставане от графика на проекта	Още в началото обхвата на проекта трябва да бъде ясно дефиниран, съгласуван между Възложителя и Изпълнителя, както трябва да бъде съгласувана и ясна процедура за въвеждане и одобряване на промени в обхвата и приоритетите.
5	Бавни процеси по оценка и вземане на решения, които биха могли да доведат до отклонения от плана и по този начин да изложат на риск цялостното изпълнение на проекта	Изоставане от графика на проекта	Възложител/ Изпълнител	15	Висока	Средна (41 - 60%)	Изоставане от графика на проекта	Мобилизиране на управленското ниво за взимане на решения. Стриктно планиране и отчитане на напредъка, за да се осигури ранно прихващане на потенциални проблеми, изискващи съгласувани решения. Съгласуване на процедури по ескалация.

№	Описание	Въздействие върху проекта	Собственик (Отговорник	Приоритет	Степен на влияние	Вероятност	Индикатор	Стратегия за смекчаване
6	Проблеми при комуникация между възложител и изпълнител – достъп до данни, разпределяне на правомощия и задължения - забавена обмяна на документи.	Изоставане от графика на проекта	Възложител/ Изпълнител	6	Умерена	Ниска (21 - 40%)	Изоставане от графика на проекта	Мобилизиране на управляващите органи на проекта. Съгласуване на план за управление на проекта и ангажимент от всички засегнати страни към процесите за управление на проекта Стриктно спазване на сроковете заложени в плана за управление на проекта
7	Проблеми при комуникация между възложител и изпълнител – неясни, непоследователни или противоречиви формулировки и изисквания към проекта	Изоставане от графика на проекта	Възложител/ Изпълнител	6	Умерена	Ниска (21 - 40%)	Изоставане от графика на проекта	Следване на заложените изисквания в техническото задание, и спазване на дефинираната и съгласувана в началото на проекта процедура за въвеждане и одобряване на промени в обхвата и приоритетите
8	Неочаквани промени в изискванията поради външни причини – закони, стандарти, оперативни решения на ръководни органи на страната, международни задължения и др.	Изоставане от графика на проекта	Възложител	6	Умерена	Ниска (21 - 40%)	Изоставане от графика на проекта	Спазване на дефинираната и съгласувана в началото на проекта процедура за въвеждане и одобряване на промени в обхвата и приоритетите
9	Възникване на проблеми при изпълнение на проекта заради трета страна в процеса на интегриране.	Изоставане от графика на проекта	Възложител/ Изпълнител	9	Умерена	Средна (41 - 60%)	Изоставане от графика на проекта	Мобилизиране на управляващите органи на проекта. Спазване на дефинираната и съгласувана в началото на проекта процедура за въвеждане и одобряване на промени в обхвата и приоритетите
10	Непостигане на критериите за приемане на проекта - непостигане технически показатели на проекта.	Недостатъчно качество на предаваните артефакти и компоненти	Изпълнител	6	Умерена	Ниска (21 - 40%)	Не приемане на предаваните	Адекватно планиране, залагане на стриктни критерии за качество на продуктите, които да бъдат оценявани еднозначно.

№	Описание	Въздействие върху проекта	Собственик (Отговорник	Приоритет	Степен на влияние	Вероятност	Индикатор	Стратегия за смекчаване
							резултати	
11	Непостигане на критериите за приемане на проекта - непостигане количествени показатели на проекта.	Недостатъчно качество на предаваните артефакти и компоненти	Изпълнител	6	Умерена	Ниска (21 - 40%)	Не приемане на предаваните резултати	Адекватно планиране, залагане на стриктни критерии за качество на продуктите, които да бъдат оценявани еднозначно.
12	Невъзможност за провеждане или сериозно затруднения за провеждане на ефективни тестове.	Изоставане от графика на проекта	Възложител/ Изпълнител	6	Умерена	Ниска (21 - 40%)	Изоставане от графика на проекта	Мобилизиране на управляващите органи на проекта. Съгласуване на план за управление на проекта и ангажимент от всички засегнати страни към процесите за управление на проекта Стриктно спазване на сроковете заложени в плана за управление на проекта
13	Отрицателни резултати от тестове, водещи до необходимост от големи промени в проекта.	Недостатъчно качество на предаваните артефакти и компоненти	Изпълнител	6	Умерена	Ниска (21 - 40%)	Не приемане на предаваните резултати	Адекватно планиране, залагане на стриктни критерии за качество на продуктите, които да бъдат оценявани еднозначно.
14	Превишаване бюджета на проекта.	Изоставане от графика на проекта	Възложител/ Изпълнител	4	Съществена	Минимална (1 - 20%)	Изоставане от графика	Следване на заложените изисквания в техническото задание, и спазване на дефинираната и съгласувана в началото на проекта процедура за въвеждане и одобряване на промени в обхвата и приоритетите
15	Кадрови проблеми – липса на специалисти с необходимата квалификация и опит	Изоставане от графика на проекта	Изпълнител	6	Умерена	Ниска (21 - 40%)	Изоставане от графика на проекта	Адекватно планиране и ясна комуникация с възложителя
16	Недостатъчно осигуряване	Недостатъчно качество	Изпълнител	6	Умерен	Ниска (21 -	Не	Адекватно планиране, залагане на

№	Описание	Въздействие върху проекта	Собственик (Отговорник	Приоритет	Степен на влияние	Вероятност	Индикатор	Стратегия за смекчаване
	на качеството по отношение на ключовите компоненти на проекта	на предаваните артефакти и компоненти			а	40%)	приемане на предаваните резултати	стриктни критерии за качество на продуктите, които да бъдат оценявани еднозначно.
17	Недостатъчно добра координация и управление на проекта	Изоставане от графика на проекта	Възложител/ Изпълнител	6	Умерена	Ниска (21 - 40%)	Изоставане от графика на проекта	Мобилизиране на управляващите органи на проекта. Съгласуване на план за управление на проекта и ангажимент от всички засегнати страни към процесите за управление на проекта
18	Бавно изпълнение на непродуктивни задачи като одобрение, одити и др., от които зависи изпълнението на продуктивните такива	Изоставане от графика на проекта	Възложител/ Изпълнител	9	Умерена	Средна (41 - 60%)	Изоставане от графика на проекта	Сроковете за изпълнение на непродуктивните задачи следва да са съгласувани предварително. Осигуряване на адекватен капацитет и ангажимент от всички заинтересовани страни. При наличието на индикации за забавяне да се потърси решение в оптимизирането на процесите и въвеждане на по-оперативни практики.
19	Недостатъчен административен капацитет за подготовка и изпълнение на бизнес и ИТ проекти	Увеличена нужда от ресурси. Изоставане от графика на проекта	Възложител	9	Умерена	Средна (41 - 60%)	Изоставане от графика на проекта, лоша комуникация	Да се увеличи числеността и квалификацията на ИТ състава.
20	Форсмажорни обстоятелства (земетресения, наводнения	Изоставане от графика на проекта	Възложител/ Изпълнител	4	Съществена	Минимална (1 - 20%)	Изоставане от графика	Ясни правила при управлението и съхранението на документацията по проекта

№	Описание	Въздействие върху проекта	Собственик (Отговорник	Приоритет	Степен на влияние	Вероятност	Индикатор	Стратегия за смекчаване
21	Забавяне при смяна на ключов експерт	Изоставане от графика на проекта	Изпълнител	6	Умерена	Ниска (21 - 40%)	Изоставане от графика на проекта	Адекватно планиране и ясна комуникация с възложителя
22	Неправилна идентификация или оценка на рискове	Изоставане от графика на проекта	Възложител/ Изпълнител	4	Съществена	Минимална (1 - 20%)	Изоставане от графика	Стриктно спазване на план за управление на риска

Относно изискванията и условията, свързани с изпълнението на предмета на настоящата процедура, ще изпълним следното:

4 МЕТОДИКА ЗА УПРАВЛЕНИЕ НА ПРОЕКТА

Целта на тази точка е да опише подхода на изпълнителя към изпълнението на обществена поръчка с предмет обособена позиция № 2 от обществена поръчка с предмет: „Последващо развитие и усъвършенстване на информационно-комуникационната среда на електронното правителство“. При изпълнението на проекта ще се следват две световно утвърдени методологии - методологията за цялостно управление на проекти на PMI (Project Management Institute) и методология за използване на единен (унифициран) процес (RUP) за разработването на софтуера. Управлението на дейностите по проекта ще следва методологията на PMI, а за разработката на софтуера ще се следва унифицирания процес RUP за разработка на софтуер, описани подробно в точка 1. Детайлно описание на изпълнението на проекта по фази и дейности е дадено в точка 2.

4.1 Организация на проекта

4.1.1 Организационна структура

При изпълнението на проекта като цяло ще се следва методологията за управление на проекти (PMI). По време на първата фаза „Планиране” се дефинират стандартите, процедурите за управление на проекта, създава организационната структура, преглежда и доразработва в пълни детайли плана за управление. Предлаганата от нас примерна организационна структура за настоящия проект е съобразена с дадената в техническото задание и е разделена на две основни нива – управленско и оперативно ниво.

Управленското ниво включва ръководител на проекта, координатор на проекта и координатор на ОП2, като представители на възложителя и ръководител на ОП2 от страна на изпълнителя. Съветът за управление на проекта наблюдава изпълнението на дейностите по проекта и контролира работата на ръководителите на проекта от двете страни. Ръководителите на проекта от двете страни са отговорни за безпроблемното изпълнение на работата по настоящия проект. Оперативно ниво – на това ниво се позиционират екипа, участващ пряко в изпълнението на проекта.

Предложената организационна структура ще бъде детайлизирана и уточнена при започване на работата по проекта, както ще бъде дефинирана и процедурата за ескалации.

4.1.2 Роли и отговорности

За управлението на проекта ще бъде създадена целева организационна структура с регламентирани роли, права и отговорности на участниците в нея. Те се осигуряват

чрез ясни документирани описания на функциите им по дейности в проекта и на изискванията към техните компетенции.

Съвет за управление на проекта

Управителният съвет се състои от следните роли:

- Ръководител на проекта от страна на Възложителя;
- Координатор на проекта от страна на Възложителя;
- Координатор на ОП2 от страна на Възложителя;
- Ръководител на проекта/екипа от страна на Изпълнителя.

Управителният съвет има следните функции:

- Представлява висшето ръководство на проекта от страна на Възложителя и Изпълнителя;
- Следи на високо ниво за напредъка на проекта и инициира контакти с УО на ОП при идентифицирана необходимост от промени в проекта, които са извън планираните обхват, срокове и качество;
- Събира се на периодични заседания, в края на всяка дейност по проекта и при необходимост;
- Осигурява форум за обсъждане и вземане на общоприето решение на всички въпроси от съществено значение за успешното изпълнение на дейността;
- Приема общия план за изпълнение на проекта.

Ръководителят на проекта от страна на Възложителя:

- Осъществява цялостното ръководство по изпълнението на проекта;
- Одобрява и подписва всички официални документи, доклади и приемо-предавателни протоколи по проекта;
- Извършва цялостен мониторинг по изпълнението на проекта.

Координаторът на проекта от страна на Възложителя:

- Отговаря за оперативната координация и изпълнението на всички заложени дейности;
- Организира, координира и следи за изпълнението на възложените задачи от координаторите на обособена позиция от страна на Възложителя;
- Работи в тясно сътрудничество с ръководителя на екипа от страна на Възложителя и координаторите на обособени позиции за своевременното разрешаване на всички възникнали проблеми и въпроси от координационно естество;
- Извършва проверки на мястото на изпълнението на всички обособени позиции;

Координатор на ОП2 от страна на Възложителя:

- Отговаря за координацията и изпълнението на съответната дейност;
- Организира периодични срещи с ръководителя на съответната обособена позиция от страна на Изпълнителя;
- Координира и организира комуникацията с партньорските организации по обособената позиция;
- Следи за напредъка по изпълнението на обособената позиция;
- Извършва проверки на мястото на изпълнението на обособената позиция;

Ръководителят на проекта от страна на Изпълнителя:

- Осъществява детайлно планиране на проекта;
- Осъществява мониторинг и контрол на оперативно ниво на напредъка на проекта;
- Осъществява мониторинг и контрол на качеството на работата, извършвана от целия проектен екип по проекта;
- Координира дейността на екипа по проекта;
- Следи за изпълнението на плана и постигане на качеството;
- Докладва на възложителя за наличието на проблеми и непредвидени рискове, за които трябва да бъдат предприети коригиращи действия;
- Участва в заседанията на Съвета за управление и докладва за напредъка на проекта;
- Отчита извършената работа от членовете на проектния екип.

Екип на изпълнителя

Екипът на изпълнителя включва ключовите експерти по проекта и други експерти, които ще участват в отделните етапи на проекта.

Необходимо условие е още от самото начало всички участници да са запознати с целите и задачите на проекта, както и процесите за неговото управление, и да участват активно в изпълнението на планираните задачи в предвидените срокове. За целта се организират специални сесии, семинари и други форми за запознаване с процедурите и методите за управление на проекта, и за цялостна подготовка на екипа за работа по проекта. Организирането на тази дейност, както и цялостното методическо обезпечаване на управлението на проекта със съответните процедури и документи, е задължение на Ръководителя на проекта.

За изпълнение на отделните дейности от обхвата на проекта ще бъдат формирани следните екипи:

- Екип за Бизнес Анализ със следните отговорности:
  - Анализ на текущо състояние;

- Моделиране на бизнес процеси;
- Дефиниране на функционалните и технически изисквания на решението
- Събиране и документиране на изискванията на Възложителя.

Ключови членове са бизнес аналитиците и експертите от страна на Възложителя.

- Екип за Проектиране и разработка със следните отговорности:
  - Специфициране на изискванията;
  - Проектиране на базата от данни;
  - Проектиране на приложението като съвкупност от модули;
  - Проектиране архитектурата на системата (работни места, комуникации, хардуер, софтуер).
  - Дефиниране на програмните интерфейси за връзка между различните модули
  - Създаване на базата от данни и реализиране на сървърната логика.
  - Програмната реализация на модулите;
  - Разработка на техническа и експлоатационна документация

Ключови членове са системен архитект, ръководител на софтуерна разработка, програмисти.

- Екип по осигуряване на качеството със следните отговорности:
  - Изготвяне на План за осигуряване на качеството;
  - Координация на дейностите по осигуряване на качеството;
  - Дефиниране на приемни тестове и тестови сценарии;
  - Подготовка на тестова среда;
  - Подбор и зареждане на тестови данни;
  - Планиране, организиране и провеждане на тестове;
  - Отчитане на резултатите от тестовете;
  - Управление на конфигурацията.

Ключови членове са експерта по осигуряване на качеството и тестовите специалисти.

- Екип за провеждане на обучението със следните отговорности:
  - Изготвяне на план за обучение;
  - Изготвяне на материали за обучение;

- Организиране и провеждане на обучение на администратори и потребители за работа със системата.

Ключов член на този екип е експерт на позиция „Ръководител на софтуерната разработка“.

4.1.3 Комуникации

По време на изпълнението на проекта експертите на Изпълнителя ще работят съвместно с представители на Възложителя, съгласно заложения план и по инициатива на някоя от страните. Използвайки съответният начин на комуникация ще се осигури ефективно разменяне на информация и комуникация както между членовете на двата екипа, така и на управленско ниво, като за целта ще се поддържат списъци с телефони, е-мейли и други средства за контакт на всички служители на двете страни, участващи в работните групи по проекта. Планирането на комуникациите определя информационните и комуникационни нужди на заинтересованите лица – кой от каква информация има нужда, кога трябва да я получи и по какъв начин. Конкретните мерки за реализация на качествена и ефективна комуникация в рамките на този проект ще се разработят по време на Фаза Планиране.

По време на изпълнението на проекта ще се използват следните методи на комуникация с цел осигуряването на акуратна и навременна информация и пълна информираност на членовете на проекта:

✓ **Срещи**

Съобразявайки се с изискванията на техническото задание планираме да организираме следните видове срещи:

- Регулярни (месечни) с цел проследяване статуса на изпълнение, с участието на ключовите експерти и членовете на съвета за управление на проекта
- При нужда – при възникване на проблеми и ескалации, с участието на ключовите експерти и членовете на съвета за управление на проекта
- Работни срещи – между експертите на работно ниво за обмен на информация и дискусии по технически въпроси

Всички срещи се документират, като формата на протоколите от срещите ще се дефинира и съгласува при стартирането на проекта. Протоколът от срещите се изготвя от страна на изпълнителя в рамките на два работни дни и се разпространява до всички участници в срещата, както и до Ръководителя на проекта от страна на Възложителя и Ръководителя на ОП2 от страна на Изпълнителя. В срок от 3 работни дни участниците в среща могат да изпратят корекции на протокола от срещата. В случай на коригиране на протокол, той отново се изпраща до всички участници. Ако не се поиска корекция до

указания срок, се смята, че всички участници в срещата са съгласни с така изготвения протокол, след което го подписват.

✓ **Кореспонденция**

Ежедневната кореспонденция ще се осъществява през е-майл, като официалните документи ще се представят в хартиен вариант и електронен вариант, съгласно изискванията.

4.1.4 Отчетност

Изпълнението на проекта ще бъде наблюдавано и документирано регулярно. Статусът на изпълнението ще се изготвя съгласно изискванията, заложени в техническото задание, което включва представяне на обобщена информация за реализираните дейности и постигнатите цели и са придружени с всички изготвени продукти и резултати. Докладите ще имат заглавна страница, която включва: име и номер на проекта, заглавие на доклада, отчетен период и дата на издаване, име на Изпълнителя.

Видове доклади за напредък

N	Отчет	Съдържание

1.	Месечни доклади	• Управленско резюме;
		• За всяка дейност от ОП2:
		  • Описание на извършената работа по дейността. Където е приложимо се прилагат и резултати от измерването на метриките за разработка;
		  • Постигнати резултати, като се опишат чрез заложените и постигнатите индикатори;
		  • Възникнали проблеми, включително забавяния или отлагане изпълнението на дейности, причини, поради които са възникнали и какви мерки са предприети за преодоляването им (ако е приложимо);
		  • Допуснати грешки, включително причини, поради които са възникнали и какви мерки са предприети за отстраняването им (ако е приложимо)
		• Актуализиран списък на рисковете;
		• Актуализиран план за изпълнение на обособената позиция, ако има промяна в него;

N	Отчет	Съдържание
		• Действия по информация и публичност.

2.	Окончателен доклад	• Управленско резюме;
		• За всяка дейност от ОП2:
		  • Описание на извършената работа по дейността. Където е приложимо се прилагат и резултати от измерването на метриките за разработка;
		  • Постигнати резултати, като се опишат чрез заложените и постигнатите индикатори;
		  • Възникнали проблеми и допуснати грешки, включително забавяния или отлагане изпълнението на дейности, причини поради, които са възникнали и какви мерки са предприети за преодоляването и отстраняването им (ако е приложимо);
		• Извлечени добри практики;
		• Препоръки за развитие в след проектния период, включително информация за бъдещата устойчивост;
		• Действия по информация и публичност;
		• Цялостна оценка на постигнатите резултати от ОП2.

3.	Доклад за гаранционната поддръжка	• Описание на извършената работа; • Възникнали проблеми и предприети мерки за преодоляването им; • Идентифицирани рискове и предприети мерки; • Действия по информация и публичност.

Месечните доклади се представят периодично до края на всеки месец. Към месечните доклади е приложено копие (включително и електронно) на изпълнената през месеца част от работата по ОП2. В края на проекта се изготвя и предава окончателен доклад,

съдържащ оценка на общото изпълнение на проекта. В края на гаранционния период се изготвя доклад за гаранционна поддръжка. Докладът се предава заедно с актуализирана версия на отчетните продукти

Всички доклади ще са съобразени с изискванията за публичност. Всички доклади се предават на български език в електронен формат и в хартиен вариант.

Отчетни продукти

Отчетните продукти се изготвят на български език и се предават в края на всяка фаза в *един* екземпляр на хартия (когато това е възможно) и в *един* екземпляр на електронен носител, включително документи, модели, програмен и изпълним код. Заедно с отчетните продукти се представя опис:

Наименование на отчетния продукт	Описание	Директория	Име на файл	Брой страници
<В съответствие с изискванията на ТЗ >	*<Описание с 1 изречение на съдържанието на отчетния продукт>*	*<Име на папката, в която е/ са файловете/ под-папките за конкретния отчетен продукт>*	*<Път до име/ имена на файловете>*	*<Само за разпечатани материали>*

Отчетните продукти са придружени с официално писмо на хартия, които се внасят в деловодството на МТИТС. Всички отчетни материали ще бъдат изготвени, спазвайки изискванията за публичност.

4.1.5 Приемане на резултатите

Изготвените от Изпълнителя отчетни резултати по отделните фази от ОП2 се предават съгласно приетия план-график в началото на проекта. Отчетните продукти се изготвят и предават на български език в 2 (два) екземпляра на хартиен и 2 (два) екземпляра на електронен носител. Възложителят ги разглежда и изготвя оценителен доклад със забележки в 7 дневен срок. До 10 дни след предаване на отчетните резултати се провежда среща за обсъждане на забележките. Приемането на резултатите се удостоверява с двустранно подписан протокол за приемане на фаза до 12 дни от предаването на резултатите. В случай на констатирани недостатъци, те се описват в протокол, като се определя и срок за отстраняването им. Окончателното приемане на работата по изпълнението на проекта се удостоверява чрез подписването на окончателен протокол. След приключване на периода на гаранционна поддръжка се подписва приемо-предавателен протокол.

4.2 Процедура за управление на промените

Управлението на промените се отнася за процедурите по контрол на промените за искания, които се считат за отклоняващи се от основните и съгласувани рамки на проекта, като тези процедури се прилагат за всички типове искания за промяна. Всички искания за промяна ще преминават през процедура за управление на промените и ще бъде поддържан регистър на управлението на промените. Исканията за промяна могат да бъдат инициирани от всеки член на екипа на възложителя или изпълнителя, като се адресират в писмена форма към ръководителя на проекта от страна на възложителя или ръководителя на екипа от страна на изпълнителя.

Следната процедура ще бъде следвана:

1. Подаване на форма за искане за промяна – искането се подава към оторизираните лица в писмен вид
2. Анализ на искането на промяна – след подаване на искането за промяна се извършва анализ и оценка , като каква промяна е необходима, какво ще е влиянието върху проекта (обхват, време, разходи, качество)
3. Оценка на цената на искането за промяна – на тази стъпка се оценява стойността на исканата промяна, влиянието ѝ върху проекта и какви усилия са необходими. Одобренията на искания за промени, касаещи установения обхват и график, се извършват от ръководителя на проекта. Промените извън тези граници се съгласуват със съвета за управление
4. Прилагане на искането за промяна – след одобрение се прилага искането за промяна, като след това се извършват проверки за качеството, за да се установи, че промените не влияят неблагоприятно на останалата част от решението
5. Поддържане на история/проследимост на исканията за промяна – всички искания за промяна се записват в регистър на исканията за промяна, като той може да съдържа следната информация: идентификация на искането, дата на иницииране, инициатор, оценяващ, статус, описание на промяната, описание на влиянието, обхват на промяната, одобряващ, статус на одобрението

След започване на проекта ще бъде съгласувана формата на искането за промяна и регистъра на исканията на промените.

4.3 Управление на качеството

Общ преглед

Качеството представлява „съвкупност от характеристиките на даден обект, които се отнасят до способността му да удовлетвори преки или косвени нужди". Управлението на качеството включва процесите, необходими за да се гарантира постигането на целите на проекта. Управлението на качеството се отнася както до управлението на проекта, така и до продукта на проекта. Управлението на качеството включва всички

дейности на цялостното управление, които определят политиката, целите и отговорностите, свързани с качеството, и ги прилага чрез средства като планиране на качеството, контрол на качеството и осигуряване на качеството:

- Планиране на качеството – определя се кои стандарти за качество са приложими към проекта и как да бъдат постигнати.
- Осигуряване на качеството – прави се периодична оценка на цялостното изпълнение на проекта, за да се даде увереност, че проектът ще постигне всички стандарти за качество, които се отнасят до него.
- Контрол на качеството – следят се конкретни резултати по проекта, за да се определи дали отговарят на съответните стандарти за качество и да се определят начини за отстраняване на причините за евентуално незадоволително изпълнение.

Основната цел на Управлението на качеството (УК) по настоящия проект, е да се гарантира, че Възложителя ще получи продукт, който отговаря на нуждите на нейните потребители и е пригоден за ползване по предвиденото му предназначение.

Успешната реализация на управлението на качеството зависи от ясната и точна комуникация в рамките на организацията и с клиентите.

При взаимодействие между страните се постига:

- Разбиране на нуждите и очакванията, за могат да бъдат адресирани при планиране на дейностите по управление на качеството
- Споделяне на опит, което може да спомогне за подобряване ефективността на системата за управление на качеството
- Своевременно сигнализиране за потенциални проблеми, които могат да повлияят на работата.

Успехът на Управлението на качеството също така зависи от ангажираността към дефинирания План за управление на качеството и спазването му от всички страни, участващи в проекта.

Основните средства на подхода за управление на качеството, който ще се използва за настоящия проект, са:

- Анализ, описание и дефиниране на практиките, средствата и последователността на дейностите, свързани с качеството, създадени за да се гарантира постигането на целите по качеството,
- Съставяне на План за управление на качеството (ПУК), който се явява основния документ на Системата за качество и включва дефиниции, ресурси и процедури за всички дейности от цялостното управление, които определят политиката, целите и отговорностите, свързани с качеството, и ги прилагат чрез средства като планиране на качеството,

контрол на качеството, осигуряване на качеството и подобряване на качеството, в рамките на системата за качество,

- Проследяване и контрол на напредъка по проекта въз основа на Плана за управление на качеството.

- Анализ и докладване на открити аномалии, забавяния, недостатъци, проблеми, както и отправяне на препоръки за отстраняването и решаването им, за да се гарантира подобряване качеството на проекта.

План за управление на качеството

Планът за управление на качеството (ПУК) ще се използва за подпомагане реализацията на проекта. Той предлага рамка, в която системата ще бъде разработена, контролирана и реализирана.

ПУК ще дефинира тази рамка по отношение на:

- Планиране на проекта
- Отчетни резултати по проекта
- Организация и отговорности по проекта
- Цели на качеството
- Дейности по контрол на качеството
- Прилагани процедури
- Управление на проекта
- Управление на конфигурацията
- Управление на промените
- Управление на рисковете

Процесът на разработване на софтуер се базира на стандартите на Unified Process (RUP). ПУК дефинира и адаптира прилагането на следните дефинирани дисциплини и резултати от RUP:

- Бизнес моделиране
- Изисквания
- Анализ и дизайн
- Реализация
- Тестване
- Внедряване, включително обучение
- Управление на конфигурацията и промените
- Управление на проекта
- Среда

Прегледи на качеството

Съществуват три вида прегледи, които по-конкретно се отнасят до бизнес въпроси, управленски въпроси и технически въпроси, както е показано на диаграмата долу. Прегледите на качеството по проекта биват:

- Преглед на бизнес въпросите (BCR): Това са срещи с цел вземане на решения и проверка дали всички програмни задачи са изпълнени

правилно, дали всички свързани рискове са били идентифицирани, анализирани и възложени на конкретни лица за съответни действия, както и с цел преглед на бизнес и финансовото състояния по програмата. Тези прегледи трябва да осигурят обективни и единодушни решения дали да се премине към следващата Фаза или не.

- Преглед на състоянието на проекта: Преглед, който се извършва периодично за проследяване напредъка и текущото състояние на графиците, разходите, плановете, основните етапи, исканията за промяна, възможностите и проблемите. Също така, тези прегледи се използват за координиране на действията между участващите подизпълнители и с цел поддържане на информираност на всички страни относно текущите програмни въпроси;

- Съвместен преглед на проекта: Преглед, който се извършва с цел оценка и проверка на състоянието на задачите на всички страни и за проследяване на напредъка, рисковете и исканията за промяна. В този преглед участват и клиентите с информация относно състоянието на техните рискове и задачи, както и за да бъдат информирани относно напредъка на проекта;

- Преглед на продуктите на работата по проекта: Всички отчетни резултати трябва да преминат технически преглед и преглед на качеството преди да бъдат доставени на клиента. Ако е подходящо, служители на клиента може да участват в този вид вътрешен преглед след предварителна уговорка с ръководството на Изпълнителя;

- Преглед на дизайна: Това е независим преглед за оценка и валидация на архитектурата и дизайна на решението. Тези прегледи трябва да се извършват от лица, които не са автори на решението, като степента на формалност зависи от изискванията на проекта;

- Независим преглед на проекта (IPR): Специален преглед, поръчан от някоя от пряко участващите страни с оглед проверка на рисковете, състоянието на клиента, финансовата цялост, техническата реализируемост и др. Този вид прегледи също така се препоръчва като начин за обмен на опит по проекти за системна интеграция.

- Преглед на етап от проекта: В края на всеки етап се прави преглед на качеството на проекта, на отчетните резултати, на процесите по доставката им и на приноса на участващите страни, като се определят и необходимите съответни действия.

- Преглед на интеграцията: Извършва се ежедневно с цел осигуряване на качеството. Изпълнителят ще изготви доклади, чрез които се

демонстрира как системата отговаря на дефинираните показатели за качество.

- Преглед на кода: Целта на този преглед е да се гарантира това, че разработения код отговаря на конвенциите, софтуерната архитектура и най-добрите практики. Изпълнителят планира да извършва прегледи на кода относно:
    - основополагащите елементи на новата технологична рамка
    - елементите, отнасящи се до сигурността
    - елементите от кода, които притежават висока степен на дефекти
    - елементи от критична важност за производителността

Показатели за качество

Ръководството и техническия персонал трябва активно да използват матрици за съобщаване на напредъка и качеството в един стабилен формат. Ако е възможно, набирането на данни за показателите трябва да е автоматизирано. В зависимост от фазата на проекта и постигнатото към момента качество, Ръководителите на проекта от страна на Възложителя и Изпълнителя решават съвместно кои показатели да се приложат и колко често да бъдат измервани.

4.4 Обучение

Целта на тази точка е да опише подхода към организацията и провеждането на обученията. Ще бъде проведено обучение на ключови потребители за работа с разработената система и администратори, които ще поддържат системата.

Обучението ще се извърши по предварително съгласуван график и ще е под формата на семинари и практически упражнения. Ще се следва следната методология:

- **Презентация** – ще се представят концептуално възможностите на функциите на разработеното решение.
- **Демонстрация** – ще се проиграят типични сценарии като се демонстрират всички стъпки от даден бизнес процес, с данни подготвени за целите на обучението.
- **Самостоятелни упражнения** – потребителите ще имат възможност самостоятелно да се запознаят с възможностите на отделните модули и начина на работа. За целите на упражненията ще бъдат разработени съвкупност от задачи, които гарантират покриване на възможностите на разработените модули и бизнес процеси.
- **Обобщения** – в края на обучението ще се прави обобщение на ключовите знания, необходими за работа със системата.
- **Въпроси и отговори** – лекциите и упражненията протичат интерактивно, като основна задача на всеки лектор е да придобие увереност, че преподавания материал е усвоен от курсистите. Стремежът е максимално пълно да се отговори на всички възникнали въпроси по време на обучението.

Обучението ще се проведе от квалифицирани специалисти, притежаващи познания по разработваните модули и с опит при провеждане на обучения.
Обхватът на обучението ще бъде допълнително детайлизиран и съгласуван с Възложителя в хода на изпълнение на проекта. Ще се разработи детайлен план за обучение за различните групи. Ще се разработи и съгласува с Възложителя учебна програма, включваща като минимум:

- Цел на обучението,
- Описание и групи потребители, за които е предназначен курсът;
- Разписание, лектор, група, място на провеждане, оборудване;
- Минимални изисквания към курсистите;
- Съдържание.

Задължение на възложителя предварително да представи списък с предвидените обучаеми. Списъкът с обучаеми е част от детайлния план за обучение. Окончателният график за обучение ще се съгласува с Възложителя в хода на изпълнение на проекта. За провеждането на ефективно и качествено обучение трябва да се извършат поредица от организационни дейности. За тази цел е необходимо да се създаде организация, в която да са включени специалисти със следните роли:

- Лектори;
- Координатор по обучението от страна на изпълнителя;
- Координатор по обучението от страна на възложителя;
- Обучаеми.

Обучението ще се провежда на български език, същото важи и за учебните материали. Подготвените учебни материали, като ръководства, презентации, копия на екрани на системата, пояснения и препратки към документацията, ще бъдат предоставени на хартиен и в електронен вид. За всяко обучение ще се изготвя присъствен списък, анкетни карти и протокол от извършено обучение. Обучението ще приключва с тест. След приключване на обучението се изготвя отчет, включващ събраните резултати от въпросниците, присъствени списъци, анализ на постигането на целите на обучението. Обучението ще се провежда в оборудвана зала с осигурен достъп до разработените модули.

4.5 Гаранционна поддръжка

Срока на гаранционната поддръжка на разработената по настоящата поръчка информационна система е 36 месеца от датата на приемането ѝ от Възложителя.

4.5.1 Ниво на техническо обслужване - service level agreement (SLA)

Предложеното Ниво на техническо обслужване - *Service Level Agreement* (SLA) съдържа първоначалните изисквания за дефиниция, обхват, класификация, организация, управление и изисквания към предоставянето на гаранционна поддръжка и обслужване на проекта.

4.5.2 Обхват на гаранционната поддръжка

В рамките на предложения гаранционен период ще се извършва поддръжка на разработеното по този договор софтуерно решение и ще се отстраняват недостатъци, свързани с него, като – открити несъответствия при функционирането на системата с изискванията на техническото задание, както и открити несъответствия между експлоатационната документация на решението и неговото функциониране. Тези дейности включват:

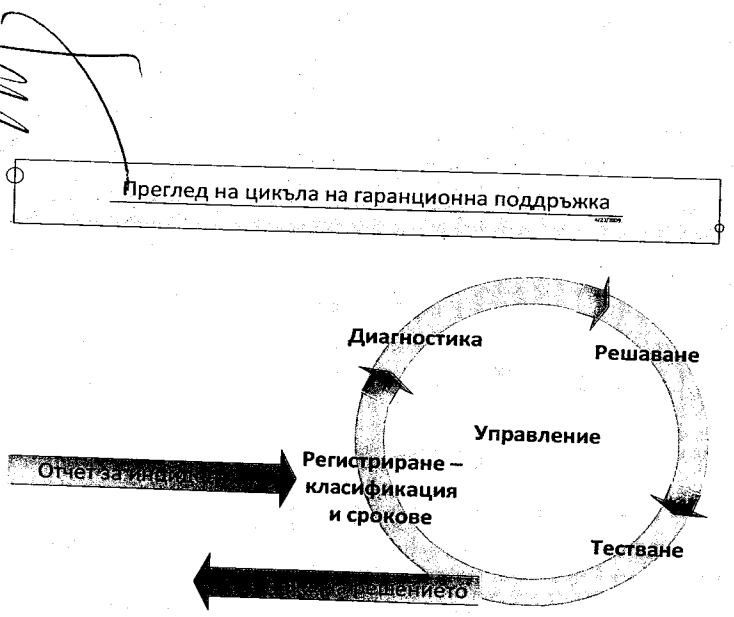
- отстраняване на дефектите, открити в софтуерните системи по проекта и съответните приложения
- корекции в следствие на грешки в системата
- Консултация за разрешаване на проблеми по предложената конфигурация на средата (операционна система, база данни, middleware, хардуер и мрежи), използвана от приложението, включително промени в конфигурацията на софтуерната инфраструктура на мястото на инсталация;
- Възстановяването на системата и данните при евентуален срив на системата, както и коригирането им в следствие на грешки в системата;
- Експертна помощ на потребителите на софтуера по телефон и електронна поща в рамките на работното време (от 9:00 до 17:30 часа всеки работен ден от седмицата)
- актуализация на документацията по проекта в съответствие с извършените коративни действия
- извършване на профилактика на разработената система на всеки три месеца.

Гаранционната поддръжка включва единствено отстраняване на проблеми и грешки в правилната работа на системата според дефинирания и приет обхват на функционалната и техническата спецификация. Гаранционната поддръжка не включва:

- разработване на нова функционалност извън обсега на проекта
- отстраняване на грешки в данните независещи от работата на системата

Тези промени могат да се извършват на база на искане за промяна, която е извън обхвата на този проект.

Процесът по отстраняване на проблеми е показан на следващата фигура:



Процесът минава през следните стъпки:

- *Регистриране и класифициране на проблемите* - След получаване на отчет за инцидент въпросът се регистрира в приложението за регистриране на проблеми, като при необходимост Възложителят може да бъде помолен да предостави допълнителна информация по проблема (като съдържание на лог-файлове, подробности по конфигурирането, тестови данни и др.) по време на всяка фаза от обработката на проблема.

- *Диагностика* - В зависимост от категорията/приоритета/влиянието/спешността на проблема се предприемат действия за локализиране на източника на проблема. Когато това е възможно, Изпълнителят ще използва собствени среди, за да възпроизведе проблема и анализира условията, които са го причинили. В някои случаи Изпълнителя може да ползва достъп до реалната/тестовата среда на Възложителя, с достъп само за четене и под наблюдение, с цел получаване на подробна информация по проблема

- *Решаване и тестване* - След като източникът на проблема бъде открит, отговорникът по инцидента докладва диагнозата и предложеното решение на Възложителя, който преглежда предложеното решение и участва в оценката му. Решението се тества за изчерпателност и съвместимост със системата. След като се тества, цялостното решение се доставя на Възложителя

4.5.3 Класификация на инцидентите:

Категория А – Сериозно влияние върху бизнес процесите - Критично влияние върху основната функционалност на приложението, то става неизползваемо;

Категория B – Съществено влияние върху бизнес процесите – Голямо влияние върху основната функционалност или критично влияние върху вторичната такава.

Категория C – Несъществено влияние върху бизнес процесите – всички останали технически проблеми

Реакция при инцидент и време за отстраняване на проблема

Категория на инцидентите	Време за реакция	Максимално време за отстраняване на проблема
A	до 1 ден	до 5 дни
B	до 3 дни	до 10 дни
C	до 5 дни	до 20 дни

Посочените срокове за реагиране на съобщения за неизправности и за отстраняване на неизправности могат да бъдат променени / регламентирани в процедура за поддръжка и обслужване, заедно с всички други въпроси във връзка с гаранционното обслужване.

Инциденти, които не са ликвидирани в договорените срокове, ще бъдат ескалирани до по-високи управленски нива в съответствие с процедурата за поддръжка и обслужване.

Отстраняването на инцидентите ще се извършва отдалечено, освен в случаите, когато естеството на проблема налага присъствие на място. Възложителят трябва да осигури възможност за отдалечен достъп и обслужване на системата.

За периода на гаранционна поддръжка ще предоставим на Възложителя и неговите структури достъп до онлайн базирана система за управление на инциденти с подходящите права за достъп според длъжностите

5 МЕТОДИКА ЗА УПРАВЛЕНИЕ НА РИСКА

Рисковете са основен фактор при управлението на един проект. По отношение на управлението на рисковете ще се базираме изцяло на PMI методологията, като ще адаптираме съответният план за нуждите на настоящия проект.

Процесът по управление на рискове представлява систематизиран процес на непрекъснато идентифициране, анализиране планиране и реакция на рисковете. Проектните рискове са събития, които при случването им могат да окажат положителен или негативен ефект върху изпълнението на проекта. За успешното управление на рисковете е необходимо да има информираност и ангажимент и от двете страни - изпълнител и възложител.

Като една от първите стъпки след започване на един проект е създаването на план за управление на рисковете, съобразен с конкретните изисквания на проекта и който е необходимо да бъде актуализиран през целия цикъл на изпълнение. Планът за управление на риска дефинира подхода, процесите за управление на риска, периодичността на разглеждане и необходимите ресурси. Като входни параметри за създаването му се използват обхвата (техническо задание, техническо предложение), плановете за управление на проекта (като управление на комуникации, на графика, на разходите), налични материали, касаещи изпълнението на проекта. Планът за управление на риска се създава съвместно от изпълнителя и възложителя. По-долу са описани основните елементи на плана за управление на риска:

Роли и отговорности

Различни роли са включени в процеса на управление на риска.

Роля	**Отговорности**
Ръководител на проекта от страна на възложителя	Одобрява предложения план за управление на рисковете и ако има промени в него Най-високо ниво при ескалация Отговаря за рисковете на ниво цялостен проект
Координатор на проекта от страна на възложителя	Отговаря за поддържането на регистъра на рискове Управлява процеса по управление на рисковете Ръководи работните срещи свързани с рисковете Отговаря за отчитането на рисковете пред ръководителя на проекта от страна на възложителя
Ръководител на проекта от страна на изпълнителя	Оказва пълно съдействие при идентифицирането, класифицирането и наблюдението на рисковете Ниво на ескалация от страна на изпълнителя
Координатори по отделните дейности от страна на изпълнителя	Отговарят за идентифицирането на рисковете за дейностите , за който отговарят и ги рапортуват на ръководител на проекта от страна на изпълнителя

Роля	Отговорности
Експерти	Участват при идентифицирането, класифицирането на рисковете в областите където участват

Описаните роли по-горе имат различно ниво на участие по време на процеса по управление на рисковете:

Стъпки	Роли и отговорности			
	Ръководител на проекта от страна на възложителя	Координатор от страна на възложителя	Ръководител на проекта от страна на изпълнителя	Експерти
Планиране на управлението на риска	А	О	О	-
Идентифициране на рисковете	И	О	О	О
Оценка на риска	И	О	О	И
Планиране на реакция при случване на даден риск	И	О	О	И
Наблюдение и контрол на рисковете	И	О	И	И

Легенда:

А - Отговорен за процеса

О - Отговаря за изпълнението на заложеното

И - Да бъде информиран

Процес по управление на риска

Процес по управление на риска се състои от следните 6 стъпки:

- *Планиране на управлението на рисковете* – определяне на подхода и планиране на дейностите, свързани с управлението на рисковете по време на проекта,

- *Идентифициране на рисковете* – определяне кой рискове могат да повлияят на проекта и документиране на техните характеристики
- *Качествен анализ на риска* – изпълнява се анализ на рисковете и условията за приоритизиране на техния ефект върху проекта.
- *Количествен анализ на риска* – оценява вероятността и последствията от рисковете и тяхното влияние върху целите на проекта в цифрово изражение
- *Планиране на реакция при случване на даден риск* – разработват процедури и техники за намаляване на влиянието на даден риск.
- *Наблюдение и контрол на рисковете* – периодично наблюдение на вече идентифицираните и приоритизирани рискове, идентифициране на нови такива, изпълнение плановете за смекчаване на рисковете и оценка на тяхната ефективност.

Планиране на управлението на риска

На тази стъпка се преглежда предложения план за управление на рисковете, извършват се промени и се одобрява окончателния вариант. При обсъждането на плана за управление на риска участват координаторът от страна на възложителя и ръководителя на проекта от страна на изпълнителя. Дефинира се процеса, планира се периодичността на наблюдение, извършва се преглед на вече идентифицираните рискове, кой да бъде информиран, дефинира се темплейт.

Идентифициране на рисковете

На този етап се идентифицират потенциалните рискове. Като първа стъпка ще се разгледат отново идентифицираните и включените в техническата оферта рискове и ще се прави периодична проверка (на всеки етап) на вече идентифицираните рискове, както дефинирането на нови такива. Идентифицирането на рисковете се осъществява на специални срещи, преглед на документацията на проекта, разговори с участниците в проекта и др. Идентифицираните рискове се описват подробно и включват информация за причината, оценка от гледна точка на вероятност и влияние, влияние върху проекта от гледна точка на разходи, време и качество. Всички идентифицирани рискове се включват в регистър на рисковете, който съдържа детайли за всички рискове, тяхната оценка от гледна точка на вероятност и влияние, отговорници, мерки за преодоляването на въздействието им и статус. Участници в процеса по идентифициране ще бъдат координаторът на страна на възложителя, ръководителя на екипа от страна изпълнителя, координаторите по отделните дейности от страна на изпълнителя и при необходимост експерти от двата екипа.

Качествен анализ на риска

При качествения анализ на риска се оценява приоритета на идентифицираните рискове, като се вземе предвид вероятността за случване на риска, степента на влияние върху проекта и взаимната връзка между рисковете. Влиянието се оценява на база обхват, разходи, време, ресурси, качество, ползи. Оценката на риска = вероятност х степен на влияние. На фигурата е показана примерна матрица на рисковете.

влияние						
	5	средно	средно	високо	високо	високо
	4	ниско	средно	средно	високо	високо
	3	ниско	средно	средно	средно	високо
	2	ниско	ниско	средно	средно	средно
	1	ниско	ниско	ниско	ниско	средно
		10%	30%	50%	70%	90%
		вероятност				

Рисковете с ниска вероятност и влияние върху проекта могат да бъдат пренебрегнати. Рисковете със средна и висока вероятност или среден и висок ефект трябва да бъдат следени и анализирани. За рисковете с висока вероятност и ефект задължително се изпълнява количествен анализ и трябва да бъдат управлявани.

Количествен анализ на риска

На тази стъпка от процеса се анализира ефекта върху проекта при случване на риска в цифрово изражение. Количественият анализ се изпълнява върху рисковете, които са приоритизирани при качествения анализ като значими за успешното изпълнение на проекта.

Планиране на реакция за предотвратяване или при случване на даден риск

На тази стъпка се планират адекватните действия, които е необходимо да се предприемат за да намали отрицателния ефект при случване на риска. Има пет типа стратегии които могат да се предприемат при планирането на реакции:

- Предпазване - избягване на риска чрез избиране на действия, които го предотвратяват. Основният фокус по време на изпълнението на проекта ще бъде насочен към този тип стратегия.
- Ограничаване - предприемане на действия, които или намаляват вероятността за появата на риска, или намаляват неговото влияние върху проекта до приемливо ниво.

- Трансфериране – специален начин на ограничаване на риска, когато рискът се трансферира на трета страна, например чрез застраховане.
- Приемане - приемане на риска поради невъзможност да се предприемат действия по смекчаване на ефектите от случване на риска.
- Овладяване - действия, които са планирани и организирани да бъдат предприети при случайно възникване на рисковата ситуация.

Наблюдение и контрол на рисковете

Тази стъпка включва периодично наблюдение на вече идентифицираните и приоритизирани рискове, идентифициране на нови такива, изпълнение плановете за смекчаване на рисковете и оценка на тяхната ефективност. Също така ще се извършва проверка, че планираните дейности имат очаквания ефект, извършва наблюдение за ранни сигнали за поява на риск, моделиране на насоки за предсказване на потенциални рискове и проверка, че цялостното управление на риска се прилага ефективно.

Списък с идентифицираните рискове е даден в точка 3. Всички рискове ще бъдат записвани в регистър.

6 МЕТОДИКА ЗА ИЗГОТВЯНЕ НА АНАЛИЗ НА РАЗРАБОТЕНИТЕ ДО МОМЕНТА КОНЦЕПЦИЯ НА БЕУ, СОФТУЕРНА АРХИТЕКТУРА НА БЕУ, ЕЛЕКТРОННА ИДЕНТИФИКАЦИЯ НА БЕУ, КАКТО И СЪЩЕСТВУВАЩИТЕ И РАЗРАБОТВАНИ В МОМЕНТА ПРОЕКТИ ЗА НОРМАТИВНА УРЕДБА НА БЕУ

В текущата част на документа ще представим накратко подхода, който ще бъде използван за изготвяне на обективен анализ на разработените до момента концепция на БеУ, софтуерна архитектура на БеУ, електронна идентификация на БеУ, както и съществуващите и разработвани в момента проекти за нормативна уредба на БеУ.

Основен източник на информация за целите на анализа ще са резултатите от изпълнението на проектите „Подобряване на административното обслужване на потребителите чрез надграждане на централните системи на електронното правителство" и „Развитие на административното обслужване по електронен път", както и посочените в техническото задание източници.

Подходът на изготвяне на анализа включва следните етапи:

- Планиране – включва дейности като формиране на екип от експерти от страна на възложителя и изпълнителя, дефиниране на заинтересованите страни, уточняване на акцента и целите на анализа, както и времева рамка. В допълнение на този етап трябва да се дефинират и предоставят всички необходими материали за извършване на анализа
- Анализ на текущо състояние – стартира с разглеждане и анализиране на разработените до момента концепция на БеУ, софтуерна архитектура на БеУ, електронна идентификация на БеУ, проекти за нормативна уредба на БеУ, организиране на интервюта с ключови експерти от страна на възложителя, провеждане на работни срещи за постигане на общо и съгласувано разбиране, както за текущото състояние и проблеми, така и по отношение на предложения подход за надежден обмен на данни и защита на ресурсите на БеУ, рискове и препоръки за развитие на технологичната рамка на БеУ. Анализът ще акцентира върху следните специфики:
    - Анализ на концепция на БеУ

      Анализът ще обхваща документите свързани с концепцията на БеУ, като ще се акцентира върху актуалността и адекватността на концепцията и степента, до която покрива настоящите и бъдещите изисквания на е-управлението. Резултатите от анализа ще се използват като входни параметри за процеса по планиране и детайлизация от последващата разработка на компонентите по настоящата поръчка.

- Анализ на софтуерна архитектура на БеУ

  Анализът ще обхваща работните версии на архитектурата на БеУ и ще вземе под внимание наличните и бъдещите компоненти на БеУ. Тук отново ще се акцентира върху актуалността и адекватността на архитектурата и степента, до която тя покрива настоящите и бъдещите изисквания на е-управлението. Резултатите от анализа ще се използват като входни параметри за процеса по планиране и детайлизация от последващата разработка на компонентите по настоящата поръчка.

- Анализ на електронна идентификация на БеУ

  Анализът ще се извърши върху документацията по разработената по предишен проект система за електронна идентификация на БеУ, като се оцени какви са възможностите за интегрирането ѝ в системата на БеУ и начините за това интегриране. Резултатите от анализа ще се използват като входни параметри за процеса по планиране и детайлизация от последващата разработка на компонентите по настоящата поръчка използващи и/или надграждащи системата за електронна идентификация на БеУ.

- Анализ на проектите за нормативна уредба на БеУ

  Ще се извърши анализ на последните версии на проектите за нормативната уредба на БеУ (закони и наредби), като фокусът ще е върху заложените нормативни изисквания, касаещи функционалността на компонентите, предмет на настоящата поръчка. Резултатите от анализа ще се използват като входни параметри за процеса по планиране и детайлизация от последващата разработка на компонентите по настоящата поръчка.

- Документиране – в резултат на проведените анализ, интервюта и работен семинар ще бъде изготвен доклад, съдържащ резултатите от анализа, предложения подход за надежден обмен на данни и защита на ресурсите на БеУ и основни рискове и препоръки за развитие на технологичната рамка на БеУ в контекста на изпълнение на настоящият проект.
- Верифициране – обсъждане и съгласуване на изготвения доклад с анализа.

7 МЕТОДИКА ЗА РЕАЛИЗАЦИЯ НА КОМПОНЕНТА ЗА ЕДНОКРАТНА АВТЕНТИКАЦИЯ НА ИС И/ИЛИ СЛУЖИТЕЛИ В АДМИНИСТРАЦИЯТА, ОСНОВАВАЩ СЕ НА SAML

7.1 Описание

Компонентът за еднократна автентикация на информационни системи и служители в администрацията (еАвт) е предназначен за нуждите на еднозначната идентификация на заявители на предоставяните от администрациите електронни услуги. Процесът се основава на издаване и валидиране на SAML токени. Такива токени ще се издават както за физически лица (в общия случай – това ще бъдат служители в администрациите), така и на информационни системи в администрациите, участващи в комплексни електронни административни услуги.

Издаването на SAML токени за физически лица ще се основава на данни от електронната им идентичност (еИД).

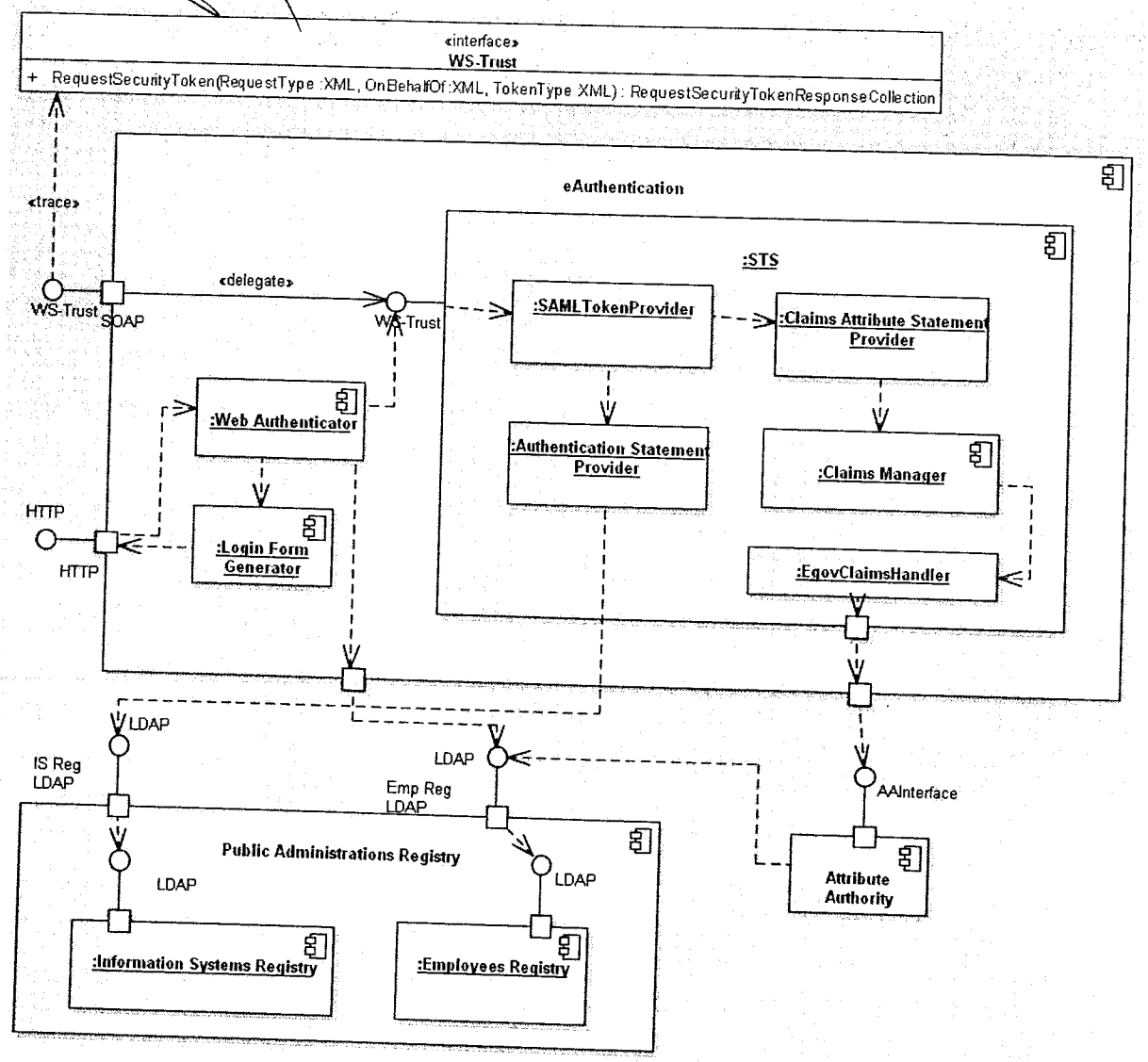
Когато заявител на услуга, като част от изпълнението на комплексна административна услуга, е информационна система, то издаването на SAML токен за тази система ще се основава на автентикация на основата X.509 сертификат. В този случай еАвт ще автентикира ИС на основата на данните в сертификата.

еАвт ще предостави функционалност за издаване на SAML 2.0 токени на физически лица на основата на SAML атестат, издаден от Валидиращия орган в процеса на идентификация на заявител. Когато заявител е гражданин, SAML токенът ще съдържа: три имена, ЕГН/ЛНЧ, секторен псевдоним. Когато заявител е служител в администрация, SAML токенът ще съдържа: три имена, ЕГН/ЛНЧ, секторен псевдоним, наименование и обектен идентификатор на администрация, както и заемана длъжност в администрацията. Данните се взимат от ПеИЗ. Служителят трябва вече да се идентифицирал, например в Портала на БеУ, предоставяйки носител на електронна идентичност или по друг начин: с потребителско име и парола. Процесът на идентификация на физически лица през Валидиращия орган е извън обхвата на дейността.

7.2 Реализация

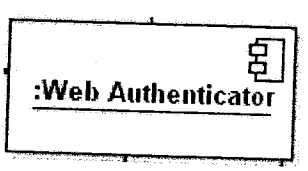
7.2.1 Компоненти

Компонентът за еднократна автентикация на информационни системи и служители в администрацията, основаващ се на SAML (еАвт), ще се състои от следните компоненти:



Фигура 1 еАвт -Логическа архитектура

7.2.1.1 Уеб Автентикатор

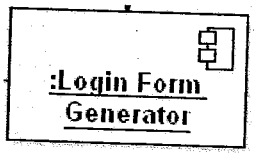


Реализира функционалността за обмен на данни със заявители на електронни услуги: служители в ДА по потребителско име и парола. Потребителските профили на служителите в ДА се поддържат в специализиран компонент „Регистър на служителите в ДА" (7.5), като валидира въведените потребителско име и парола, автентикира заявителя през „Регистър на служителите в ДА", генерира WS-Trust заявка за издаване на SAML токен от специализиран компонент „

Компонент за издаване на токени (STS)" (0), получава SAML токен от STS, кодира токена според протокола SAML Redirect Binding и го връща на заявилата система. Такава система в общия случай ще бъде „Порталът за електронно управление в Република България" (ПБеУ).

Компонентът ще бъде реализиран като JBoss SEAM EJB компонент.

7.2.1.2 Генератор на форма за логин



Генерира уеб форма за логин с полета за въвеждане на потребителско име и парола. На основата на тези данните от тази форма компонентът „

Уеб Автентикатор" (7.2.1.1) автентикира служителя в ДА.

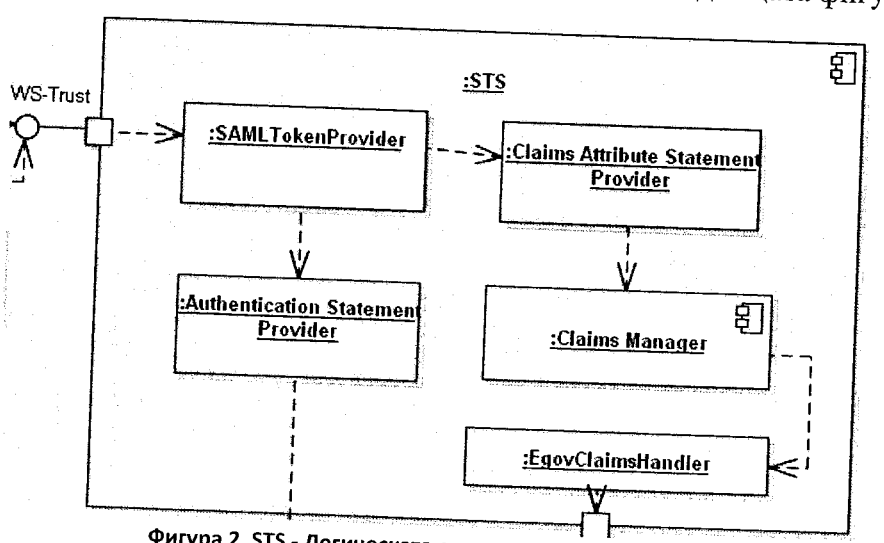
Ще бъде реализирана по технологията Facelet с JBoss RichFaces.

7.2.1.3 Компонент за издаване на токени (STS)

Компонентът за издаване на токени (STS) е отговорен за издаване на идентифициращите SAML токени. Компонентът ще реализира стандартен протокол WS-Trust.

Ще бъде реализиран с продукта Apache CXF STS.

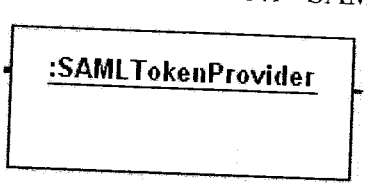
Логическата структура на компонента STS е изобразена на следващата фигура:



Фигура 2 STS - Логическата структура на компонента

STS се състои от следните компоненти:

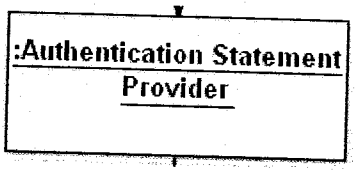
7.2.1.3.1 SAML Token Provider



STS позволява регистрирането на токен генератори (token providers), към които се делегира създаването на токен от определен вид, например: SAML v1, SAML v2, Kerberos и др. За нуждите на БеУ видът на издаваните токени ще бъде SAML версия 2 (SAML V2).

Ще се реализира с Java програмен интерфейс, предоставен от продукта Apache CXF STS.

7.2.1.3.2 Authentication Statement Provider



Този генератор е отговорен за автентикирането на заявители на електронни услуги, когато те са информационни системи.

За повече подробности виж сценарий „Автентикация на информационна система като потребител на електронни услуги".(7.2.2.3).

Ще се реализира с Java програмен интерфейс, предоставен от продукта Apache CXF STS.

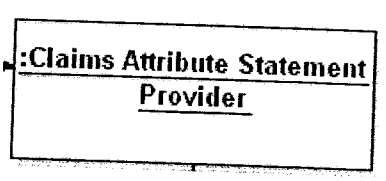
За целта ще бъде разработен Java клас реализиращ интерфейса:
`org.apache.cxf.sts.token.provider.AuthenticationStatementProvider`

Пример за такъв клас е даден по-долу:

```
public class EGovAuthenticationStatementProvider implements
AuthenticationStatementProvider{
...

    public AuthenticationStatementBean getStatement(TokenProviderParameters
providerParameters){

            ...

            // Логика за автентициране на служител през Регистър на
служителите в ДА

            ...
}
...
}
```

Логиката за връзка с компонента „Регистъра на служителите в ДА" по LDAP протокол ще бъде реализирана в тялото на метода `getStatement()`.

7.2.1.3.3 Claims Attribute Statement Provider



Този генератор е отговорен за извличането на допълнителна информация за служител в ДА, за който се заявява издаването на SAML токен. Допълнителната информация ще се използва за нуждите на оторизация на достъпа до защитени ресурси в БеУ, както и за целите на одита в средата на електронно управление. Видът на допълнителната информация се описва със специални claim елементи, дефинирани в XML схемата на протокола WS-Trust.

Тъй като видът на издавания токен е SAML, подходът за включване на claim елементите в токена е да се генерират SAML *attribute statements*, които съдържат стойностите на допълнителната информация. Това ще се постигне с реализацията на специално разработен *java* клас `AttributeStatementProvider`, който се регистрира в обекта `SAMLTokenProvider`.

Ще се реализира като java клас с Java програмен интерфейс, предоставен от продукта Apache CXF STS.

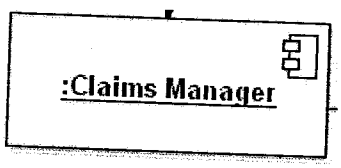
Ще бъде разработен Java клас реализиращ интерфейса:
`org.apache.ws.security.saml.ext.bean.AttributeStatementBean`

Пример за такъв клас е даден по-долу:

```
public class EGovClaimsAttributeStatementProvider implements
AttributeStatementProvider{
    ...
    public AttributeStatementBean getStatement(TokenProviderParameters
providerParameters){
        ...
        // Логика за иницииране на генераторите за claims елементите
         EgovClaimsHandler ch= new EgovClaimsHandler(…);
        ClaimCollection cc= ch.retrieveClaimValues(…);
        ...
    }
...
}
```

1. За повече подробности виж сценарий „Автентикация на служител в ДА с потребителско име и парола".(7.2.3.1).

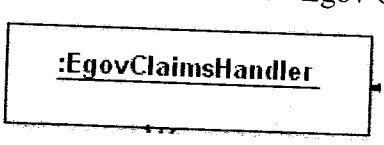
7.2.1.3.4 Claims Manager



Този компонент поддържа списък от един или повече Claims Handler – това са компонентите, които извличат същинските стойности за атрибутите на субекта.

За нуждите на изпълнението на проекта ще има разработен един такъв компонент: Egov Claims Handler.

7.2.1.3.5 Egov Claims Handler



Този компонент е отговорен за извличането на допълнителна информация за физическо лице, за което се заявява издаването на SAML токен. Допълнителната информация ще се използва за нуждите на оторизация на достъп до защитени ресурси в БеУ, както и за целите на одита.

Egov Claims Handler ще извлича допълнителни атрибути за субекта от „Справочник за атрибути" (7.7) с LDAP заявки.

Ще се реализира с Java програмен интерфейс, предоставен от Apache CXF STS.

Ще бъде реализиран като Java клас, реализиращ интерфейса:
`org.apache.cxf.sts.claims.ClaimsHandler`

Пример за такъв клас е даден по-долу:

```
public class EGovClaimsHandler implements ClaimsHandler{
...
    public ClaimCollection retrieveClaimValues(
       RequestClaimCollection claims, ClaimsParameters parameters) {
          ...
          // Логика за извличане на атрибути от Справочник за атрибути
          ...
}
...
}
```

Логиката за връзка с компонента „Справочник за атрибути" по SOAP протокол ще бъде реализирана в тялото на метода `getStatement()`.

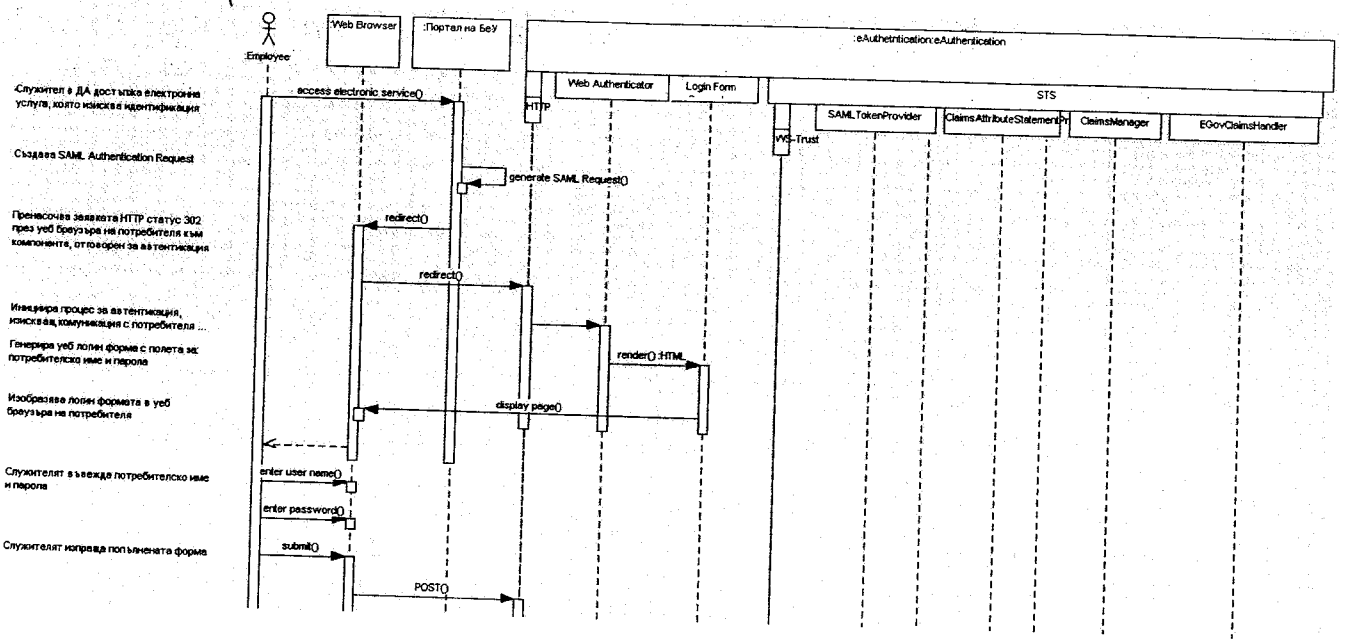
7.2.2 Сценарии

За илюстрация на използването на еАвт ще разгледаме следните хипотетични сценарии:

- Автентикация на служител в ДА с потребителско име и парола (т. 7.2.2.1);
- Автентикация на физическо лице -гражданин или служител в ДА с еИД (т. 7.2.2.2);
- Автентикация на информационна система като потребител на електронни услуги (т. 7.2.2.3).

7.2.2.1 Автентикация на служител в ДА с потребителско име и парола

Първата част на сценария е изобразена на следващата диаграма:



Сценарият започва, когато служител в ДА иска да използва определена електронна услуга, предоставяна от администрация.

През уеб браузъра на своето работно място той отваря страницата Портала на БеУ (ПБеУ), където са достъпни описанията електронните административни услуги, предоставяни от администрациите.

Тъй като изпълнението на електронната услуга изисква идентификация с *потребителско име и парола*, ПБеУ трябва да генерира заявка за автентикация на служителя. Заявката е предназначена за еАвт.

ПБеУ знае HTTP адреса на еАвт, който за целите на техническото предложение приемаме, че е `https://eauth.egov.bg/authenticate`, и съставя следното SAML съобщение:

```
<AuthnRequest ID="…"
 Version="2.0"
 IssueInstant="2014-08-05T12:30:12"
 ProtocolBinding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-POST"
 ProviderName="portal.egov.bg"
 AssertionConsumerServiceURL="http://portal.egov.bg/eservices/index/...."
>
 <Issuer>portal.egov.bg</Issuer>
 <NameIDPolicy AllowCreate="true"
  Format="urn:oasis:names:tc:SAML:1.1:nameid-format:unspecified"
 />
</AuthnRequest>
```

Съдържанието в елемента `AuthnRequest [@AssertionConsumerServiceURL]` определя адреса, на който еАвт трябва върне генерирания SAML токен – в случая това е адреса на ПБеУ.

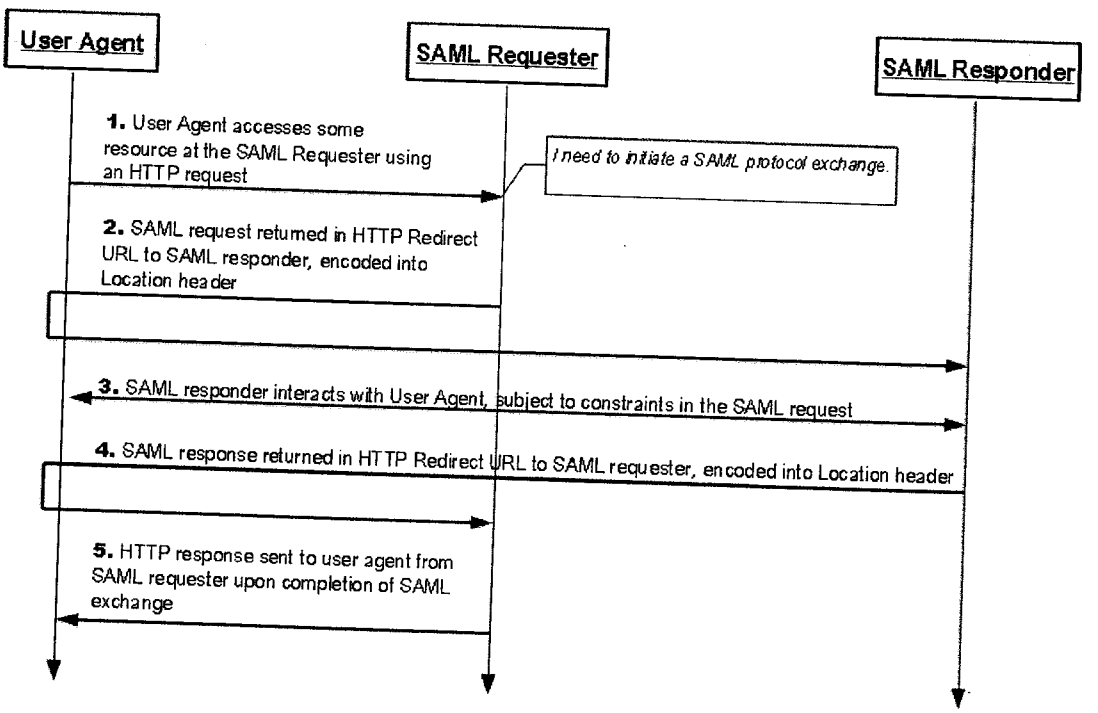
Съдържанието в `AuthnRequest/Issuer` определя коя система е заявила автентикацията. Това е също така ПБеУ.

Така генерираната заявка се кодира и се добавя след параметъра `SAMLRequest` в HTTP.

Примерно съдържание на генерираната от ПБеУ HTTP заявка е дадено долу:

```
HTTP/1.1 302 Object Moved
Date: 21 Jul 2014 09:50:05 GMT
Location:
https://eauth.egov.bg/authenticate?SAMLRequest=fVFdS8Mw...
&SigAlg=http%3A%2F%2Fwww.w3.org%2F200%2F09%2Fxmldsig%23rsa-sha1&Signature=...
Content-Type: text/html; charset=utf-8
```

Последователността от действия следва стандартния процес SAML Redirect Binding. Схематично описание на процеса е изобразено на следващата фигура:



Фигура 3 Използване на SAML redirect binding

еАвт получава заявката и валидира информацията в нея.

еАвт проверява в локалния си кеш дали вече няма издаден валиден SAML токен за съответния служител.

Ако има такъв токен, този токен се връща, с което процесът по автентикация завършва.

При липса на токен в локалния кеш, еАвт делегира изпълнението към компонента „Генератор на форма за логин" (7.2.1.2).

Генераторът създава логин-форма с полета за потребителско име и парола.

Забележка: По време на изпълнение на проекта ще се прецени дали са необходими допълнителни данни за успешна идентификация на служителя освен потребителско име и парола.

Генерираната форма се изпраща към браузъра на служителя.

Служителят попълва изискваните данни и изпраща формата.

След получаване на данните еАвт стартира процес по автентикация на служителя.
Този процес е изобразен на следващата диаграма:

Служителят изпраща попълнената форма

Инициира процес за автентикация на служителя

Системата автентикира служителя през LDAP

Системата определя вида на потребителя: служител или гражданин

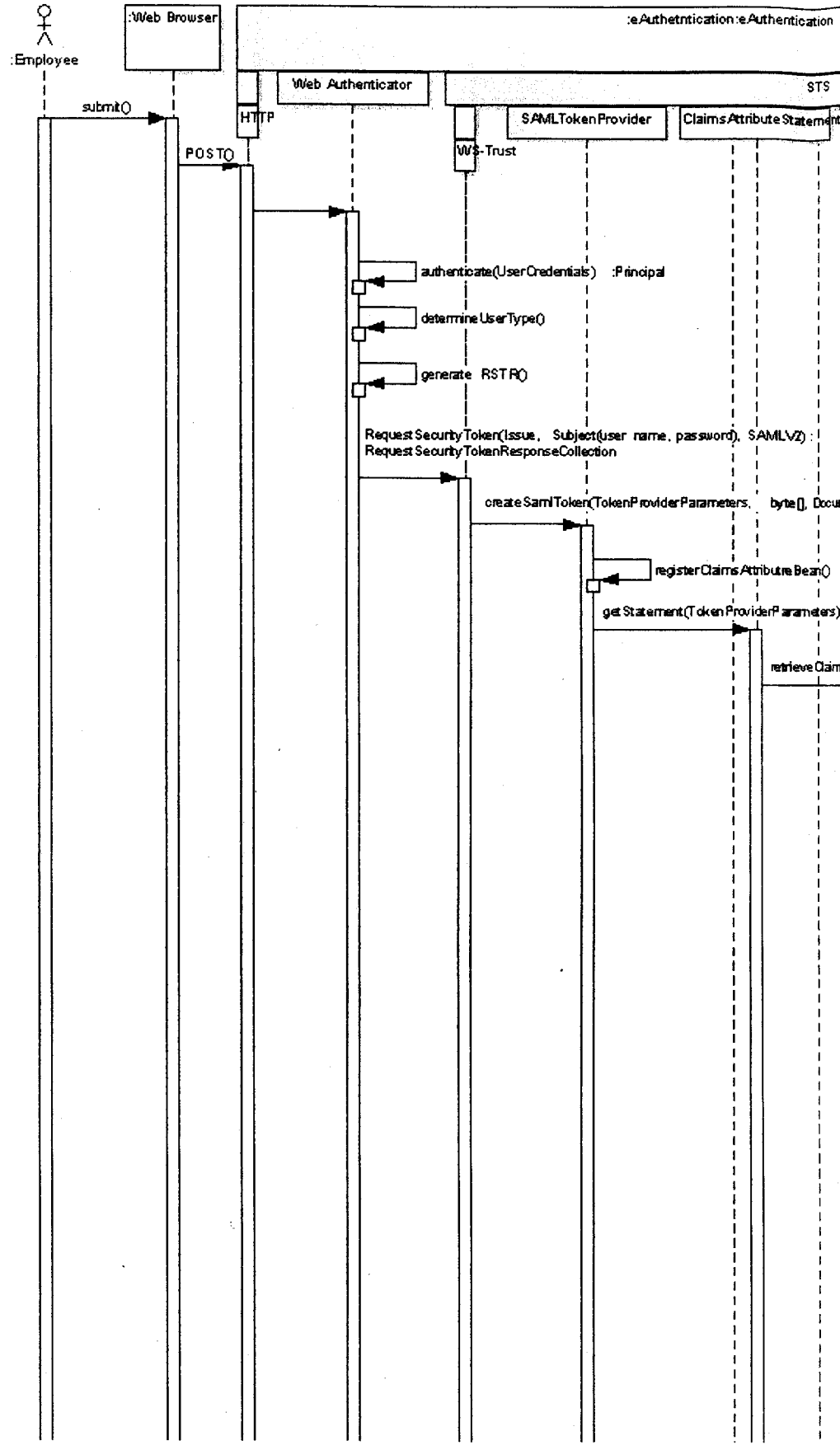
Системата генерира Request Security Token Request с предоставените потребителско име и парола на служителя, вид на токена SAMLv2 и claim елементи за допълнителната информация за служител

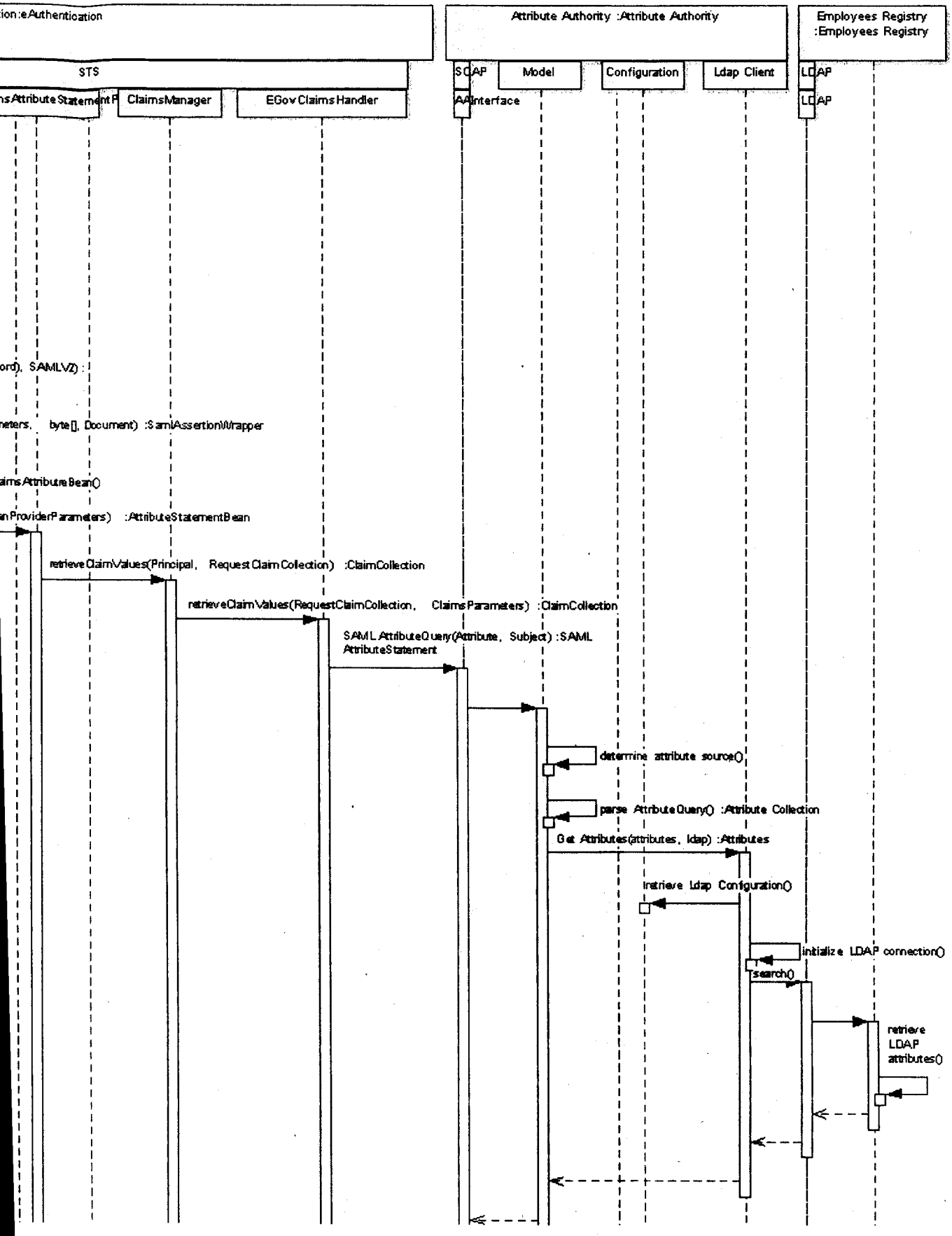
Инициира се издаването на SAML токен

Логиката за извличане на допълнителната информация е в тялото на метода getStatement()

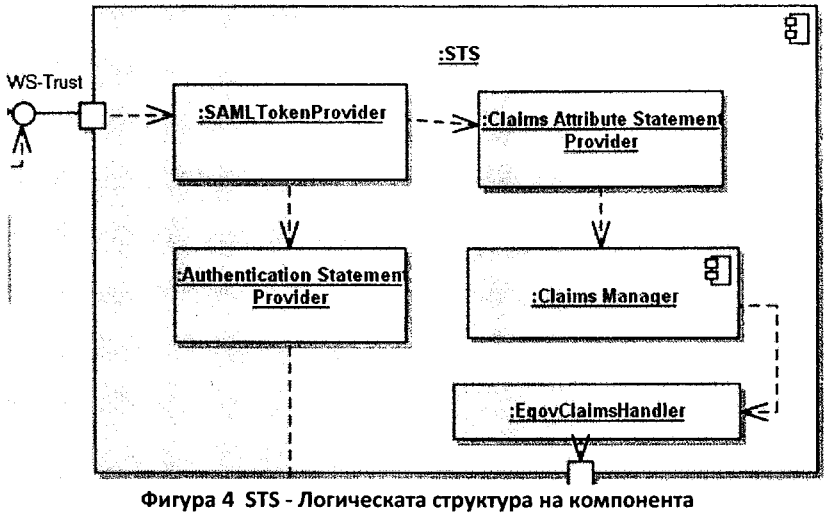
За всеки claim елемент

Генерира се заявка SAML Attribute Query




ion:eAuthentication
Attribute Authority :Attribute Authority
Employees Registry :Employees Registry
STS
SOAP
Model
Configuration
Ldap Client
LDAP
ClaimsManager
EGov Claims Handler
AAInterface
LDAP
ord), SAMLV2) :
neters, byte[], Document) :SamlAssertionWrapper
ims Attribute Bean()
nProviderParameters) :AttributeStatementBean
retrieveClaimValues(Principal, RequestClaimCollection) :ClaimCollection
retrieveClaimValues(RequestClaimCollection, ClaimsParameters) :ClaimCollection
SAML AttributeQuery(Attribute, Subject) :SAML AttributeStatement
determine attribute source()
parse AttributeQuery() :Attribute Collection
Get Attributes(attributes, ldap) :Attributes
retrieve ldap Configuration()
initialize LDAP connection()
search()
retrieve LDAP attributes()

Действителната автентикация се извършва от STS-компонент (Security Token Service), който има следната логическа структура:



Фигура 4 STS - Логическата структура на компонента

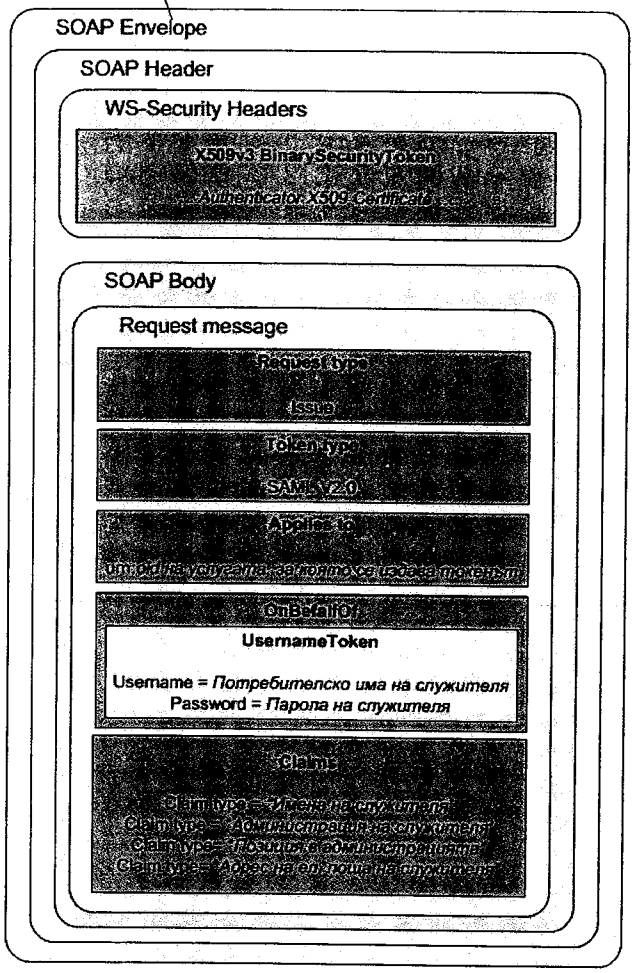
Описание на компонентите има в т. 7.2.1.3 (Компонент за издаване на токени (STS).

За служителите в ДА се изисква допълнителна информация, която да бъде включена в генерирания SAML токен: три имена, администрация, позиция в администрацията и адрес на електронна поща. Тази информация е необходима за оторизация на достъпа до електронната услуга (процесът е разгледан подробно 8. Методика за реализация на компонент за електронна оторизация, позволяващ дефиниране на гъвкави правила за разрешаване или ограничаване на достъпа до системни ресурси).

Стандартът WS-Trust предоставя тази възможност чрез специални елементи *Claim* , които са дефинирани в XML схемата на стандарта.

Тези елементи описват данни, които трябва да бъдат включени в генерирания SAML токен. *Claim* елементите се генерират от компонента „Уеб Автентикатор".

Компонентът „Уеб Автентикатор" генерира WS-Trust заявка със следната примерна логическа структура:



X.509 токенът служи за идентификация на автентикатора пред STS

Вид на токена, който трябва да се генерира: SAML версия 2.0

Данните на заявителя: потребителско име и парола, въведени в логин формата

Описание на допълнителни данни за заявителя, които трябва да присъстват в SAML токена

Фигура 5

STS получава заявката за издаване на токен и валидира данните в нея.

По стойностите на username и password в елемента `OnBehalfOf/UsernameToken` компонентът STS автентикира служителя през „Регистър на служителите в Държавната" (РСДА) (7.5). Данните за всички служители в ДА в Република България ще се поддържат от РСДА. Основна част от РСДА е LDAP регистър. Търсенето в РСДА ще се осъществява по протокол LDAP.

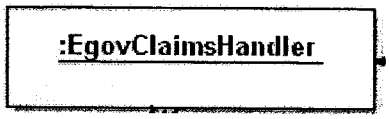
STS проверява в РСДА за наличието на служител с предоставените потребителско име и парола и извлича пълно име DN (distinguished name) на служителя, например:

```
CN=Иван Иванов, OU=Дирекция, O=МТИТС, CN=dir, DC=egov, DC=bg
```

След успешно валидиране на данните, STS обработва всеки един *Claim* елемент.

Извличането на допълнителни характеристики (атрибути) се осъществява от специализиран компонент „Справочник за атрибути" (виж 7.7)

За целта в STS се конфигурира т.нар. Claims Handler.



Този компонент е отговорен за извличането на допълнителна информация за физическо лице, за което се заявява издаването на SAML токен. Допълнителната информация ще се използва за нуждите на оторизация на достъп до защитени ресурси в БеУ, както и за целите на одита.

Egov Claims Handler ще извлича допълнителни атрибути за субекта от „Справочник за атрибути" (7.7) с LDAP заявки.

Ще се реализира с Java програмен интерфейс, предоставен от Apache CXF STS.

Ще бъде реализиран като Java клас, реализиращ интерфейса:
`org.apache.cxf.sts.claims.ClaimsHandler`

Пример за такъв клас е даден по-долу:

```
public class EGovClaimsHandler implements ClaimsHandler{
...
    public ClaimCollection retrieveClaimValues(
       RequestClaimCollection claims, ClaimsParameters parameters) {
          ...
          // Логика за извличане на атрибути от Справочник за атрибути
          ...
}
...
}
```

Логиката за връзка с компонента „Справочник за атрибути (СпрА)" (7.7) по SOAP протокол ще бъде реализирана в тялото на метода `getStatement()`.

Claims Handler генерира заявка към (СпрА) със следното примерно съдържание:

```
<samlp:AttributeQuery …>
<saml:Subject>
   <saml:NameIdentifier
Format="urn:oasis:names:tc:SAML:1.1:nameid-format:X509SubjectName"
NameQualifier="http://dir.egov.bg/...">
   CN=Иван Иванов,OU=Дирекция,O=МТИТС,DC=dir,DC=egov,DC=bg
</saml:NameIdentifier>
</saml:Subject>

<saml:Attribute
      NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:uri"
      Name="urn:oid:2.5.4.42"
      FriendlyName="givenName">
    </saml:Attribute>

<saml:Attribute
      NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:uri"
      Name="urn:oid:2.5.4.4"
      FriendlyName="sureName">
    </saml:Attribute>
```

```
<saml:Attribute
      NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:uri"
      Name="urn:oid: 1.2.100.2.1.1"
      FriendlyName="department">
    </saml:Attribute>

    ...

    <saml:Attribute
      NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:uri"
      Name="urn:oid:1.3.6.1.4.1.1466.115.121.1.26"
      FriendlyName="mail">
    </saml:Attribute>
</samlp:AttributeQuery>
```

Описанието на данните, които СпрАтр трябва да извлече, са описание в елементите saml:Attribute. В горния случай това са:

- Име → „givenName" (urn:oid:2.5.4.42)
- Фамилия →„sureName" (urn:oid:2.5.4.4)
- Отдел → „department" (urn:oid: 1.2.100.2.1.1)
- Адрес на електронна поща → „mail" (urn:oid:1.3.6.1.4.1.1466.115.121.1.26)

СпрА приема заявката и валидира данните в нея.

СпрА поддържа връзката между вид на атрибута, зададен с неговия обектен идентификатор (например urn:oid:2.5.4.42) и информационна система, която е отговорна за поддържането на информацията за съответния атрибут. Информационните системи са собственост на държавни агенции –администратори на първични данни

Компонентът СпрА ще позволява дефинирането на 2 вида интерфейси:

- LDAP;
- База данни.

Тъй като атрибутът `oid:2.5.4.42` е дефиниран в LDAP схема, съответната информационна система, която е отговорна за поддържането му, е компонентът „Регистър на служителите в Държавната Администрация" (виж 7.5).

СпрА генерира LDAP заявка по данните е елемента `saml:NameIdentifier` (това е `CN=Иван Иванов,OU=Дирекция,O=МТИТС,DC=dir,DC=egov,DC=bg`) и извлича съответните атрибути.

СпрА генерира отговор на заявката AttributeQuery. Отговорът има следното примерно съдържание:

```
<saml:Assertion
```

```
    xmlns:saml="urn:oasis:names:tc:SAML:2.0:assertion"
    …
    ID="..."
    Version="2.0"
    IssueInstant="2014-07-17T20:31:41">
    <saml:Issuer>...</saml:Issuer>
    <ds:Signature>...</ds:Signature>

    <saml:Subject>
      <saml:NameID
        Format="urn:oasis:names:tc:SAML:1.1:nameid-format:X509SubjectName">
        CN=Иван Иванов,OU=Дирекция,O=МТИТС,DC=dir,DC=egov,DC=bg
      </saml:NameID>
      ...
    </saml:Subject>
…

    <saml:AttributeStatement>
      <saml:Attribute
        xmlns:x500="urn:oasis:names:tc:SAML:2.0:profiles:attribute:X500"
        x500:Encoding="LDAP"
        NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:uri"
        Name="urn:oid:2.5.4.42"
        FriendlyName="givenName">
        <saml:AttributeValue
          xsi:type="xs:string">Ivan</saml:AttributeValue>
      </saml:Attribute>

      <saml:Attribute
        xmlns:x500="urn:oasis:names:tc:SAML:2.0:profiles:attribute:X500"
        x500:Encoding="LDAP"
        NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:uri"
        Name="urn:oid:2.5.4.4"
        FriendlyName="sureName">
        <saml:AttributeValue
          xsi:type="xs:string">Ivanov</saml:AttributeValue>
      </saml:Attribute>

      <saml:Attribute
        xmlns:x500="urn:oasis:names:tc:SAML:2.0:profiles:attribute:X500"
        x500:Encoding="LDAP"
        NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:uri"
        Name="urn:oid:1.2.6.1.4.1.1466.115.121.1.26"
        FriendlyName="mail">
        <saml:AttributeValue
          xsi:type="xs:string">ivan.ivanov@mtits.bg</saml:AttributeValue>
      </saml:Attribute>

    </saml:AttributeStatement>
  </saml:Assertion>
```

STS ClaimsHandler получава отговора на заявката.

След успешно завършване на процеса на автентикация на служителя и събиране на допълнителната информация, STS генерира SAML v2 токен. Примерна структура на токена е дадена по-долу:

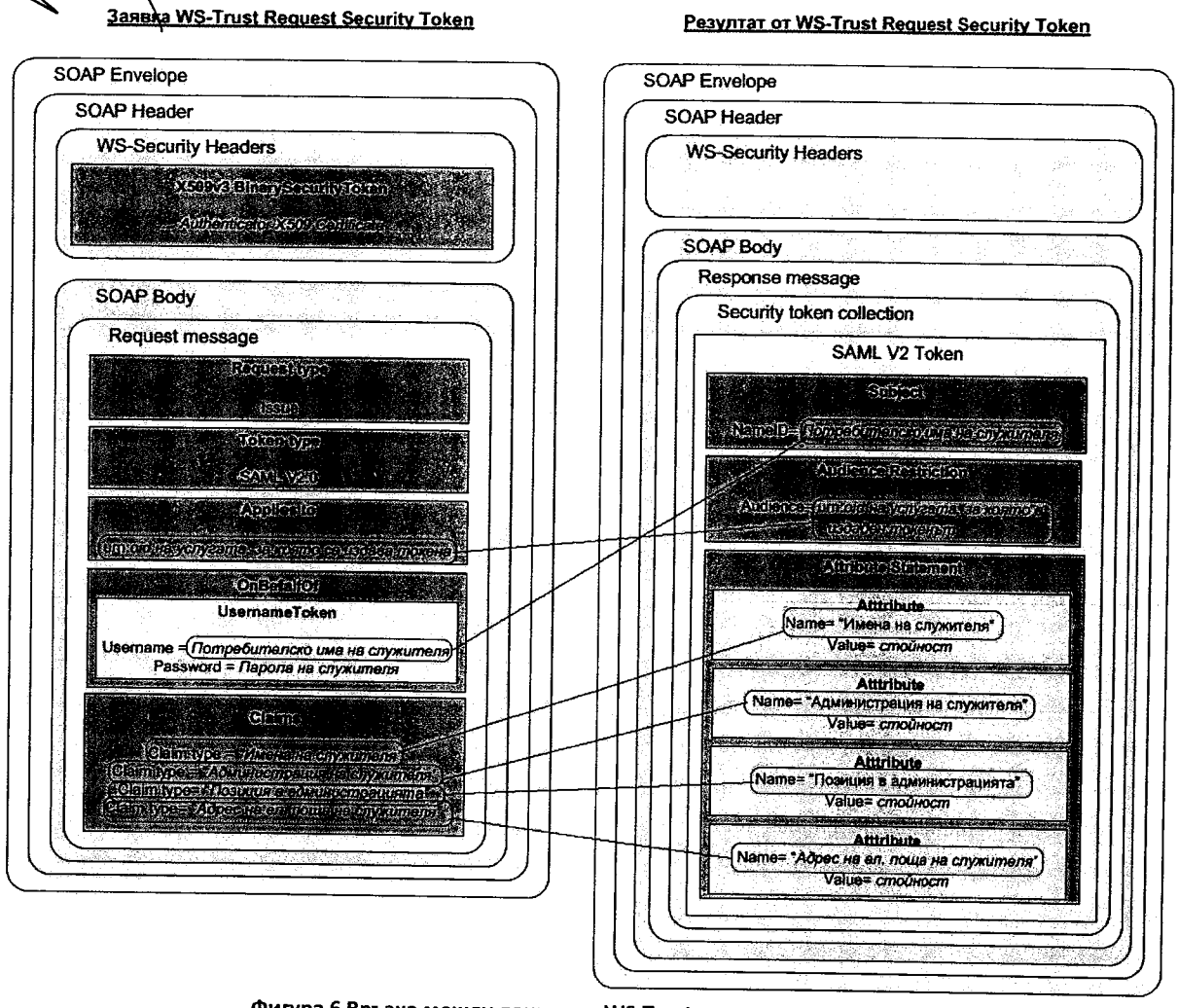
```
<saml:Assertion
```

```
xmlns:saml="urn:oasis:names:tc:SAML:2.0:assertion"
xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
ID="…"
Version="2.0"
IssueInstant="2014-08-05T09:22:05">
<saml:Issuer>http://eauth.egov.bg/...</saml:Issuer>

<ds:Signature xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
   ...
</ds:Signature>

<saml:Subject>
    <saml:NameID
          Format="urn:oasis:names:tc:SAML:2.0:assertion#X509SubjectName"
          Name="urn:oid:2.5.4.45"
          FriendlyName="uniqueIdentifier">
        ЕГН/ЛНЧ
    </saml:NameID>

    <saml:SubjectConfirmation
      Method="urn:oasis:names:tc:SAML:2.0:cm:sender-vouches">
      <saml:SubjectConfirmationData
         InResponseTo="aaf23196-1773-2113-474a-fe114412ab72"
         Recipient="https://portal.egov.bg...."
         NotOnOrAfter="2004-12-05T09:27:05"/>
      </saml:SubjectConfirmation>
</saml:Subject>

<saml:Conditions
  NotBefore="2014-10-05T09:17:05"
  NotOnOrAfter="2014-10-05T19:27:05">
  <saml:AudienceRestriction>
    <saml:Audience>https://www.administrationA.egov.bg</saml:Audience>
  </saml:AudienceRestriction>
</saml:Conditions>

<saml:AuthnStatement
  AuthnInstant="2014-08-05T09:22:00"
  SessionIndex="…">
```

```
        <saml:AuthnContext>
          <saml:AuthnContextClassRef>
            urn:oasis:names:tc:SAML:2.0:ac:classes:Password
          </saml:AuthnContextClassRef>
        </saml:AuthnContext>
      </saml:AuthnStatement>
      <saml:AttributeStatement>

      <saml:Attribute
        xmlns:x500="urn:oasis:names:tc:SAML:2.0:profiles:attribute:X500"
        x500:Encoding="LDAP"
        NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:uri"
        Name="urn:oid:2.5.4.42"
        FriendlyName="givenName">
          <saml:AttributeValue xsi:type="xs:string">
             Ivan
          </saml:AttributeValue>
      </saml:Attribute>

      <saml:Attribute
         xmlns:x500="urn:oasis:names:tc:SAML:2.0:profiles:attribute:X500"
         x500:Encoding="LDAP"
         NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:uri"
         Name="urn:oid:2.5.4.4"
         FriendlyName="sureName">
          <saml:AttributeValue xsi:type="xs:string">
             Petrov
          </saml:AttributeValue>
      </saml:Attribute>
  </saml:Assertion>
```

STS пакетира връща на компонента „Уеб Автентикатор" генерирания SAML токен

Връзката между данните в WS-Trust заявката и резултата е изобразена на следващата фигура:



Фигура 6 Връзка между данните в WS-Trust заявката и резултата

Уеб Автентикаторът получава резултата от заявката, извлича SAML токена, подписва го и генерира HTTP отговор към браузъра на потребителя. Заявката е с HTTP статус 302 (redirect). Токенът е кодиран и се намира в параметъра `SAMLResponse`:

```
HTTP/1.1 302 Object Moved
Date: 21 Jan 2004 07:00:49 GMT
Location:
http://portal.egov.bg/eservices/index/....?SAMLResponse=fVFNa4Q...
&SigAlg=http%3A%2F%2Fwww.w3.org%2F200%2F09%2Fxmldsig%23rsa-
sha1&Signature=...
Content-Type: text/html; charset=utf-8
```

Получавайки тази заявка, уеб браузърът на служителя пренасочва заявката към ПБеУ.

ПБеУ извлича генерирания SAML токен, създава при необходимост потребителска сесия, съхранява токена и обвързва всяка следваща заявка на конкретния потребител с този SAML токен до изтичане на периода му на валидност (елементи `NotBefore` и `NotOnOrAfter`) и отчитайки предназначението му (елемент

<saml:AudienceRestriction>). Примерно съдържание на споменатите елементи е дадено долу:

```
<saml:Conditions
  NotBefore="2014-10-05T09:17:05Z"
  NotOnOrAfter="2014-10-05T9:27:05Z">
  <saml:AudienceRestriction>
    <saml:Audience>https://www.administrationA.egov.bg</saml:Audience>
  </saml:AudienceRestriction>
</saml:Conditions>
```

7.2.2.2 Автентикация на физическо лице -гражданин или служител в ДА с еИД

Сценарият започва, когато гражданин или служител в ДА иска да използва определена електронна услуга, предоставяна от администрация и разполага с носител на еИд.

През уеб браузъра на своето работно място той отваря страницата в Портала на БеУ (ПБеУ), където са достъпни описанията на електронните административни услуги, предоставяни от администрациите.

Тъй като изпълнението на електронната услуга изисква идентификация с *еИД*, ПБеУ трябва да генерира заявка за автентикация на служителя. Заявката е предназначена за Валидиращия орган (ВО).

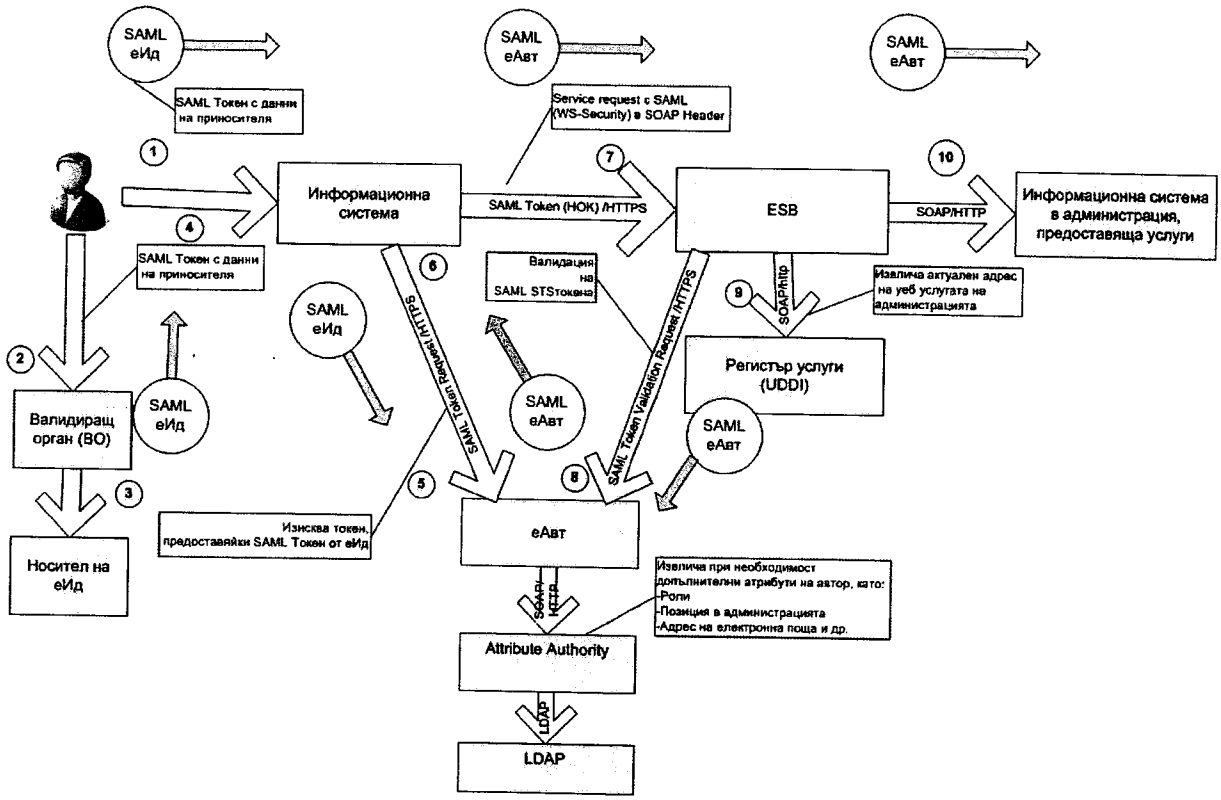
Валидиращият орган:

- инициира комуникация с четеца на карти, с който разполага работното място на потребителя;
- стартира java аплет, записан в картата
- през аплета изчита данните за електронната идентичност (еИд)
- валидира данните
- издава SAML атестат, съдържащ следните данни:
  - Три име на кирилица;
  - Три имена на латиница;
  - ЕГН/ЛНЧ;
  - Дата на раждане;
  - Секторен псевдоним.
- изпраща атестата на ПБеУ.

Издаденият от ВО SAML атестатът е само първа стъпка в процеса на автентикация на потребителя. Действителната автентикацията, състояща се от издаването на токен,

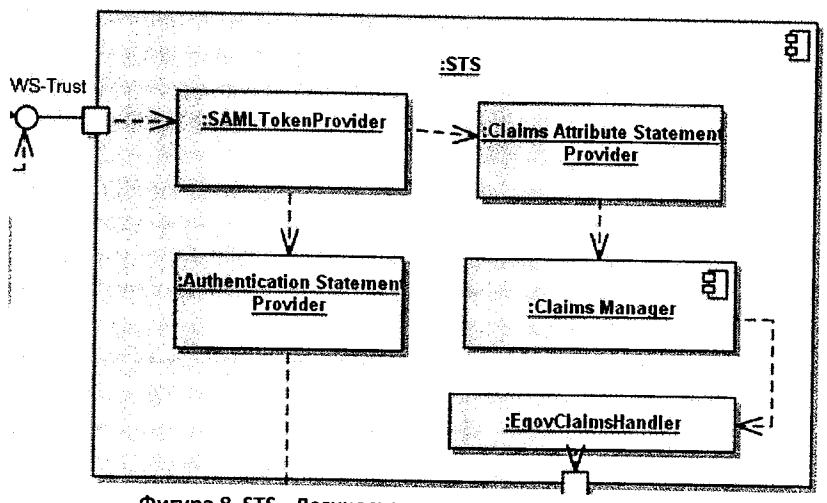
който може в последствие да бъде валидиран от всяка една заинтересована информационна система, се извършва от STS-компонент (Security Token Service).

Процесът на трансформиране на SAML атестатът в SAML токен и използването на двата вида токени е изобразен на следващата фигура:



Фигура 7 Процес на трансформация на SAML токени

Действителната автентикация се извършва от STS-компонент (Security Token Service), който има следната логическа структура:



Фигура 8 STS - Логическата структура на компонента

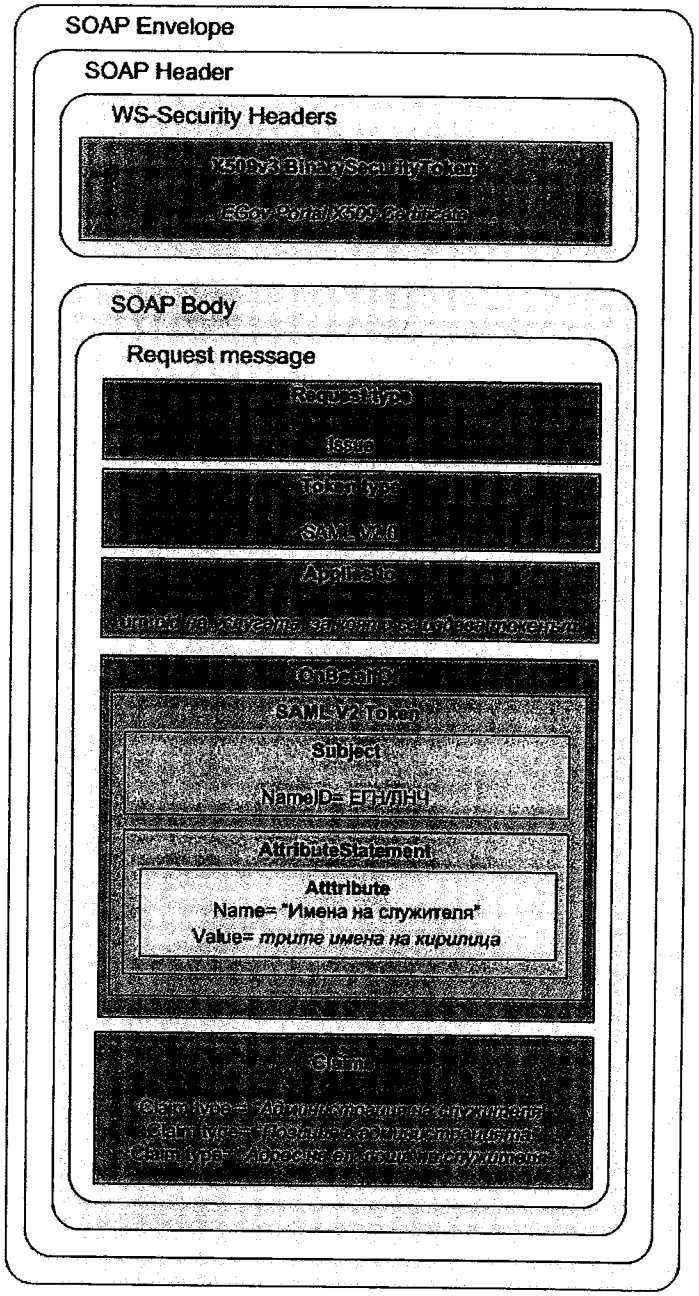
Описание на компонентите има в т. 7.2.1.3 (Компонент за издаване на токени (STS)).

За служителите в ДА се изисква допълнителна информация, която да бъде включена в генерирания SAML токен: три имена, администрация, позиция в администрацията и адрес на електронна поща. Тази информация е необходима за оторизация на достъпа до електронната услуга (процесът е разгледан подробно в 8. Методика за реализация на компонент за електронна оторизация, позволяващ дефиниране на гъвкави правила за разрешаване или ограничаване на достъпа до системни ресурси).

Стандартът WS-Trust предоставя тази възможност чрез специални елементи *Claim* , които са дефинирани в XML схемата на стандарта.

Тези елементи описват данни, които трябва да бъдат включени в генерирания SAML токен. *Claim* елементите се генерират от компонента „Уеб Автентикатор".

Когато заявител на услугата е служител в ДА, ПБеУ ще генерира WS-Trust заявка със следната примерна логическа структура:



X509 сертификат на ПБеУ.

По този сертификат еАвт ще идентифицира ПБеУ.

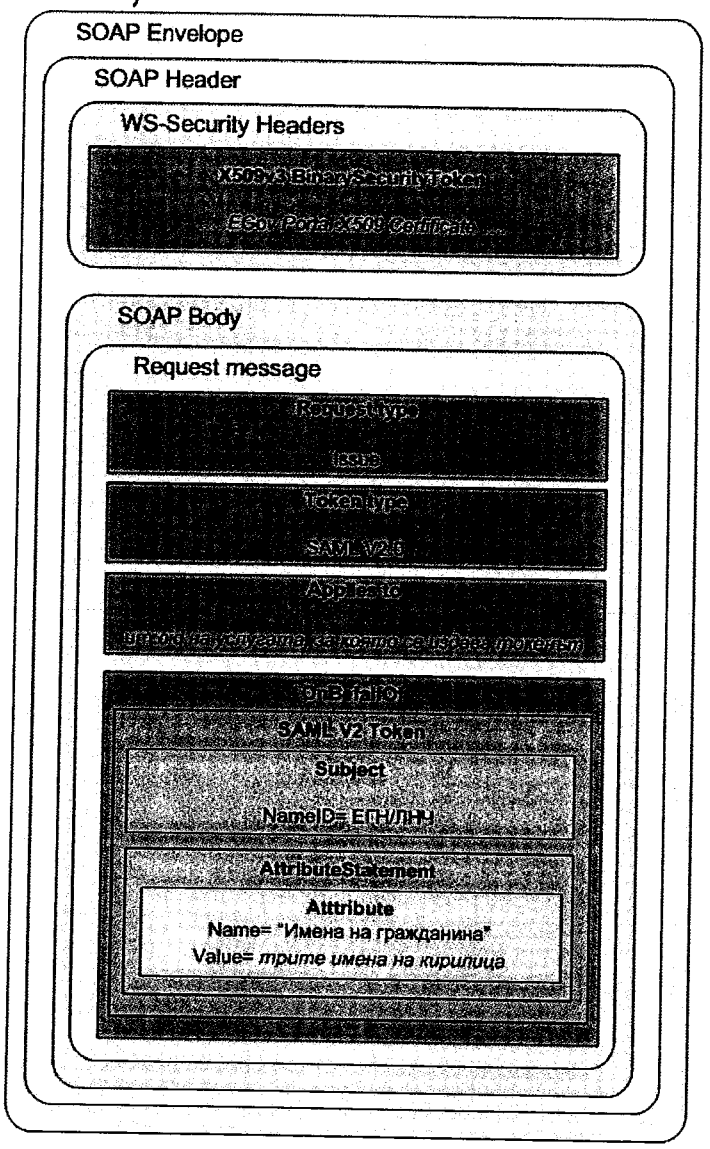
Вид на токена, който трябва да се генерира: SAML версия 2.0

SAML атестат, генериран от Валидиращ Орган и съдържащ следните данни за служител:

- Идентификатор на физ. лице (ЕГН, ЛНЧ)
- 3 имена на служителя

Описание на допълнителни данни за заявитея, които трябва да присъстват в SAML токена

Когато заявител на услугата е гражданин, ПБеУ ще генерира WS-Trust заявка със следната примерна логическа структура:



X509 сертификат на ПБеУ.

По този сертификат еАвт ще идентифицира ПБеУ.

Вид на токена, който трябва да се генерира: SAML версия 2.0

SAML атестат, генериран от Валидиращ Орган и съдържащ следните данни за гражданин:

- Идентификатор на физ. лице (ЕГН, ЛНЧ)
- 3 имена

STS получава заявката за издаване на токен и валидира данните в нея.

STS извлича от хедъра на SOAP съобщението X509 сертификата на ПБеУ.

STS валидира данните в сертификата спрямо локално си хранилище със сертификати.

Забележка: Ако в ИТ средата на МТИТС има разгърната среда PKI, то съхранението на сертификатите ще бъде централизирано и валидирането им ще може да става по стандартизиран начин, например по XKMS (XML Key Management Specification) уеб услуга. Това ще бъде изяснено в хода на изпълнение на проекта.

Когато заявител е служител в ДА:

По стойностите в SAML атестата в елемента `OnBehalfOf` компонентът STS автентикира служителя през „Регистър на служителите в Държавната" (РСДА) (7.5). Данните за всички служители в ДА в Република България ще се поддържат от РСДА. Основна част от РСДА е LDAP регистър. Търсенето в РСДА ще се осъществява по протокол LDAP.

STS проверява в РСДА за наличието на служител с предоставения идентификатор: ЕГН или ЛНЧ и извлича пълното име DN (distinguished name) на служителя, например:

```
CN=Иван Иванов, OU=Дирекция, O=МТИТС, CN=dir, DC=egov, DC=bg
```

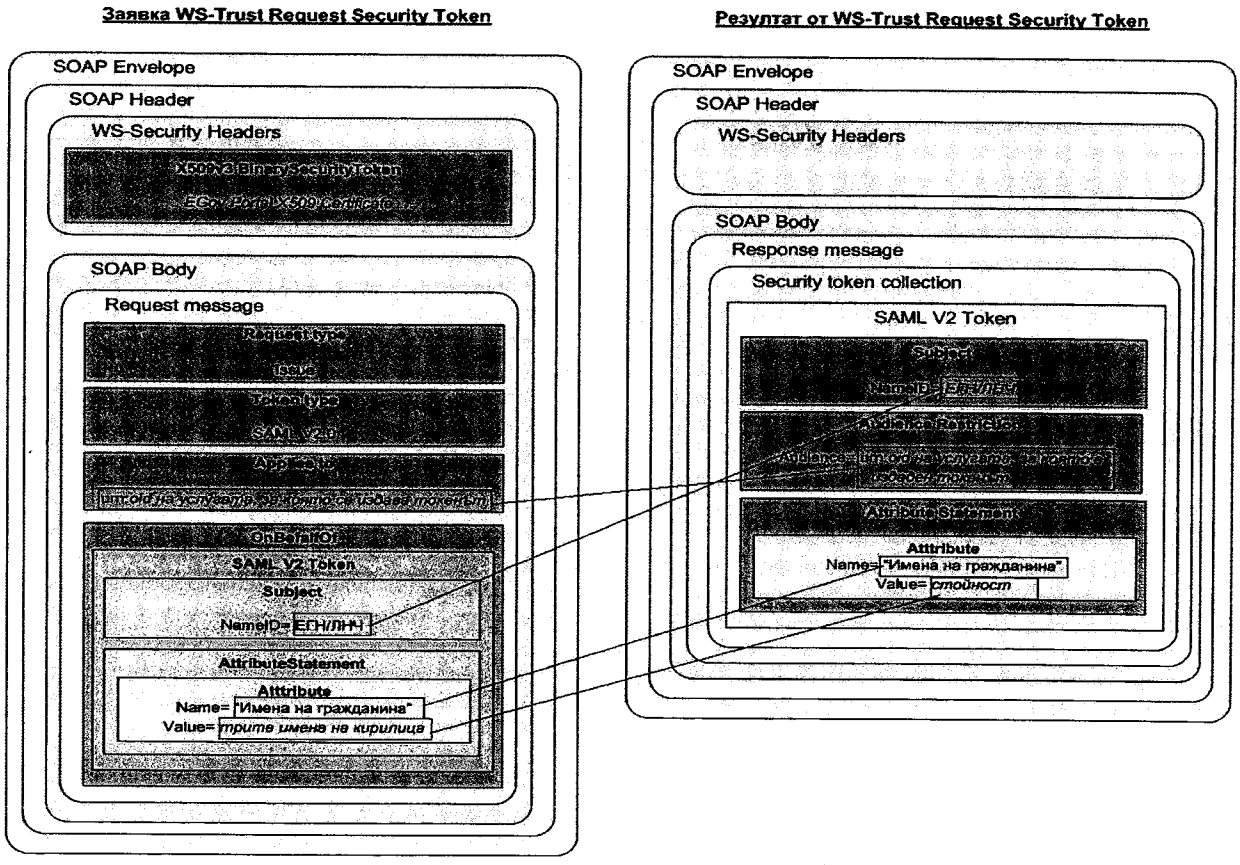
След успешно валидиране на данните, STS обработва всеки един *Claim* елемент.

Извличането на допълнителни характеристики (атрибути) се осъществява от специализиран компонент „Справочник за атрибути" (виж 7.7).

22. По-нататък процесът е аналогичен на процеса „Автентикация на служител в ДА с потребителско име и парола" (7.2.2.1).

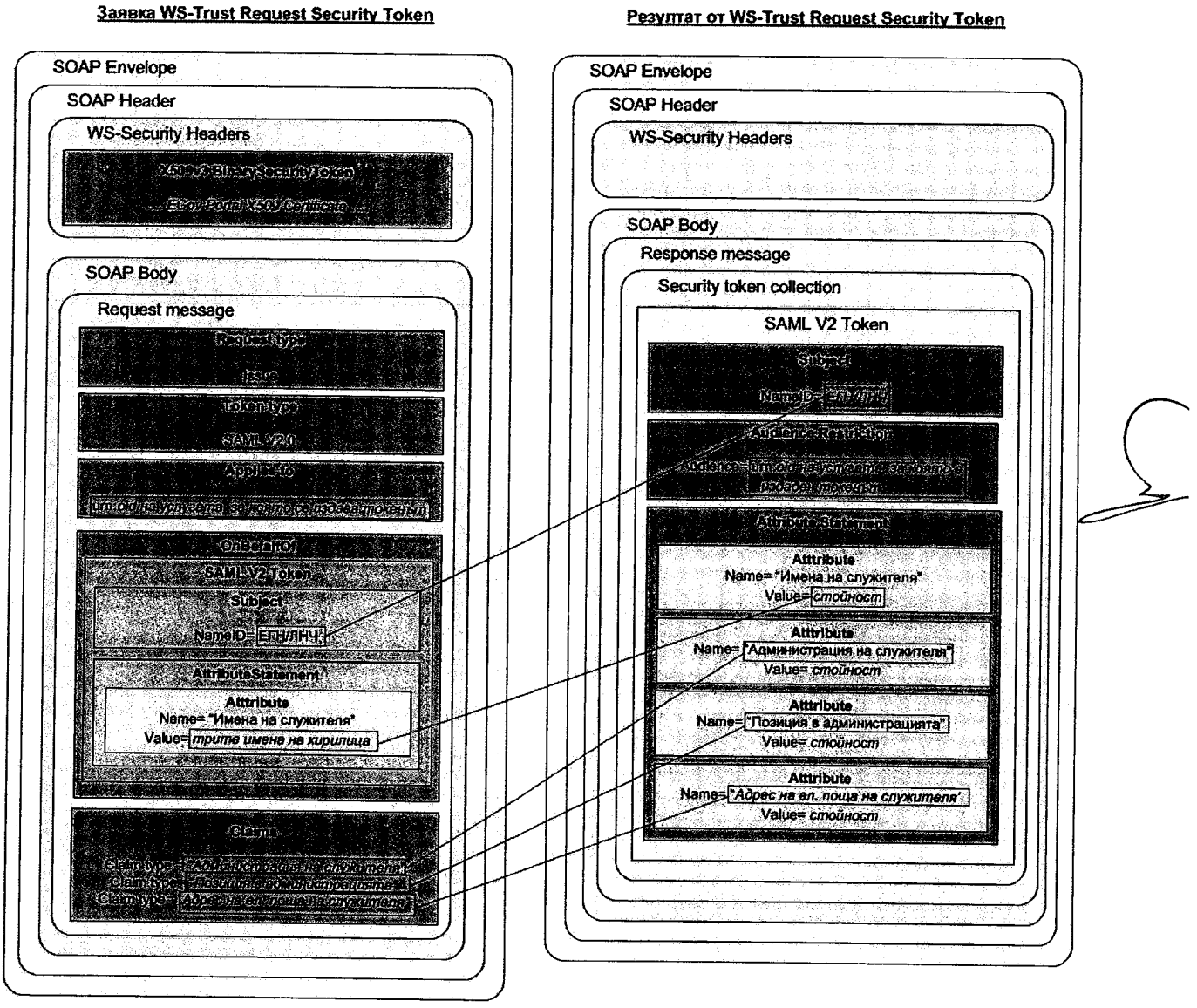
Когато заявител е гражданин:

- еАвт пакетира генерирания SAML токен в WS-Trust отговор и връща на компонента ПБеУ резултата от заявката:
- Връзката между данните в WS-Trust заявката и резултата е изобразена на следващата фигура:



Когато заявител е служител в ДА:

- еАвт пакетира генерирания SAML токен в WS-Trust отговор и връща на компонента ПБеУ резултата от заявката:
- Връзката между данните в WS-Trust заявката и резултата е изобразена на следващата фигура:



еАвт изпраща WS-Trust съобщението на ПБеУ.

ПБеУ извлича генерирания SAML токен, създава при необходимост потребителска сесия, съхранява токена и обвързва всяка следваща заявка на конкретния потребител с този SAML токен до изтичане на периода му на валидност (елементи `NotBefore` и `NotOnOrAfter`) и отчитайки предназначението му (елемент `<saml:AudienceRestriction>`). Примерно съдържание на споменатите елементи е дадено долу:

```
<saml:Conditions
  NotBefore="2014-10-05T09:17:05"
```

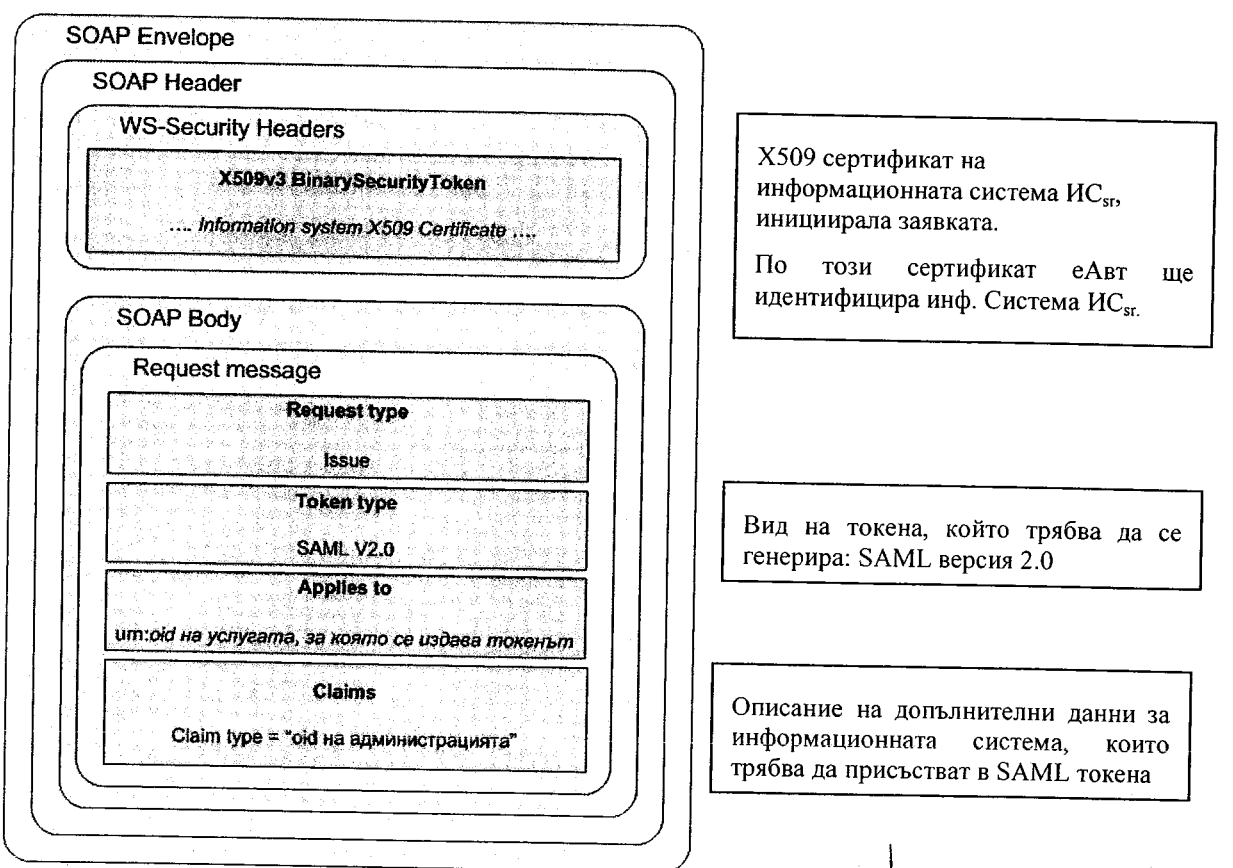
```
    NotOnOrAfter="2014-10-05T19:27:05">
    <saml:AudienceRestriction>
      <saml:Audience>https://www.administrationA.egov.bg</saml:Audience>
    </saml:AudienceRestriction>
  </saml:Conditions>
```

7.2.2.3 Автентикация на информационна система като потребител на електронни услуги

Този сценарий се отнася за информационни системи, които в автоматичен режим генерират заявка за използване на уеб услуга (обозначена по-нататък с $ИС_{sr}$), предоставяна от друга информационна система (обозначена с $ИС_{sp}$) , като част от изпълнение на процес на комплексно административно обслужване.

В този случай заявяващата система $ИС_{sr}$ трябва да включи в SOAP заявката токен, по който информационната система – доставчик на електронна услуга $ИС_{sp}$ да може да автентикира обмена на данни.

$ИС_{sr}$ генерира заявка към еАвт за издаване на SAML токен според WS-Trust протокола. Заявката ще има следната примерна логическа структура:



еАвт получава заявката, валидира данните в нея.

еАвт извлича от хедъра на SOAP съобщението X509 сертификата на информационната система.

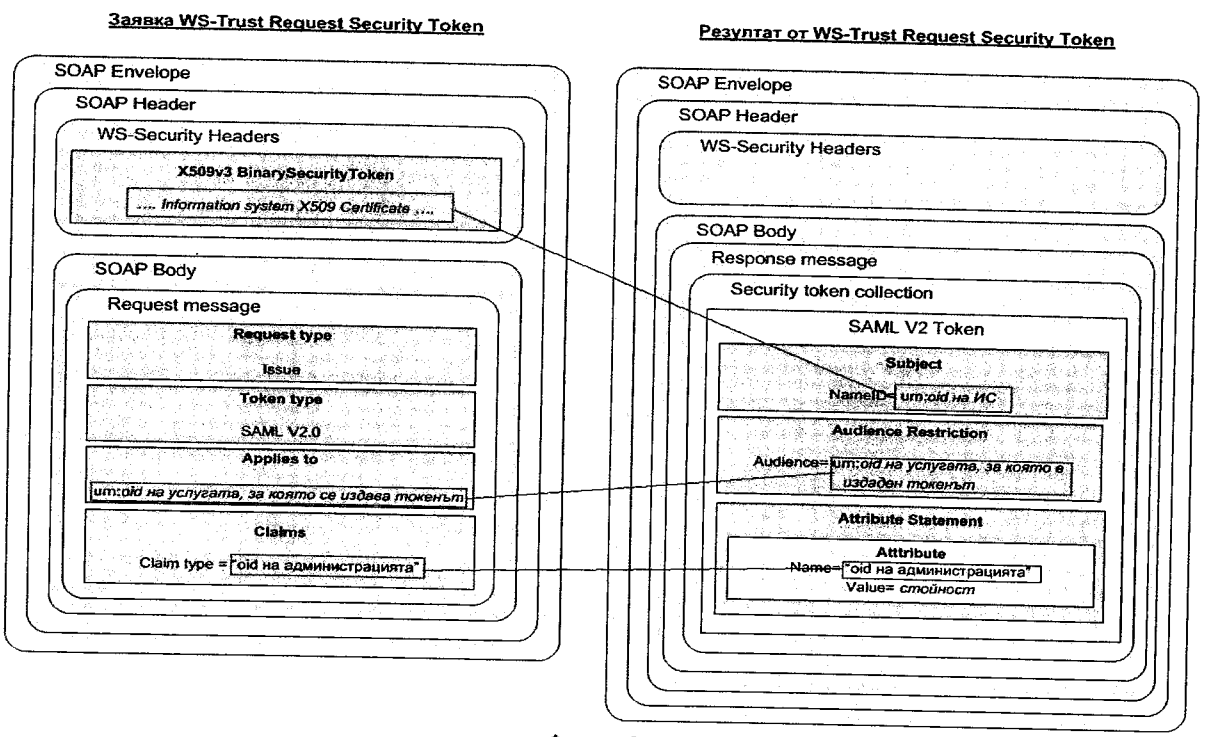
еАвт валидира данните в сертификата спрямо локално си хранилище със сертификати.

Забележка: Ако в ИТ средата на МТИТС има разгърната среда PKI, то съхранението на сертификатите ще бъде централизирано и валидирането им ще може да става по стандартизиран начин, например по XKMS (XML Key Management Specification) уеб услуга. Това ще бъде изяснено в хода на изпълнение на проекта.

Ако валидирането на данните в сертификата на информационната система е успешно, то информационната система се счита за автентикирана и STS обработва всеки един *Claim* елемент. Данните за администрация се извличат от Регистъра на администрациите в ДА по LDAP протокол.

Извличането на допълнителни характеристики (атрибути) се осъществява от специализиран компонент „Справочник за атрибути" (виж 7.7).

еАвт пакетира информацията в SAML токен, генерира WS-Trust отговор и връща на компонента $ИС_{sr}$ резултата от заявката. Връзката между данните в WS-Trust заявката и резултата е изобразена на следващата фигура:



Фигура 9

$ИС_{sr}$ извлича генерирания SAML токен, евентуално съхранява токена за последващи извиквания.

$ИС_{sr}$ извиква уеб услугата от $ИС_{sp}$.

SAML токенът се предава според спецификацията WS-Security SAML Token Profile

$ИС_{sp}$ получава заявката, извлича SAML токена и валидира:

- периода му на валидност (елементи `NotBefore` и `NotOnOrAfter`);
- предназначението на токена (елемент `<saml:AudienceRestriction>`).

Примерно съдържание на споменатите елементи е дадено долу:

```
<saml:Conditions
  NotBefore="2014-10-05T09:17:05"
  NotOnOrAfter="2014-10-05T19:27:05">
  <saml:AudienceRestriction>
    <saml:Audience>https://www.administrationA.egov.bg</saml:Audience>
  </saml:AudienceRestriction>
</saml:Conditions>
```

$ИС_{sp}$ може при необходимост да валидира издадения SAML токен. STS предоставя тази възможност чрез протокола WS-Trust Validation Binding. Примерна заявка за валидиране на SAML токен ще има следния вид:

```
<wst:RequestSecurityToken xmlns:wst="...">
    <wst:TokenType>
       http://docs.oasis-open.org/ws-sx/ws-trust/200512/RSTR/Status
    </wst:TokenType>
    <wst:RequestType>
        http://docs.oasis-open.org/ws-sx/ws-trust/200512/Validate
    </wst:RequestType>
    <wst:ValidateTarget>
          SAML токен
    </wst:ValidateTarget>
</wst:RequestSecurityToken>
```

Елементът `/wst:RequestSecurityToken/wst:ValidateTarget` съдържа SAML токена, който трябва да бъде валидиран.

еАвт получава заявката, проверява валидността на SAML токена и връща следния резултат:

```
<wst:RequestSecurityTokenResponse xmlns:wst="..."  >
   <wst:TokenType>...</wst:TokenType>
   <wst:RequestedSecurityToken>...</wst:RequestedSecurityToken>
...
   <wst:Status>
      <wst:Code>...</wst:Code>
      <wst:Reason>...</wst:Reason>
   </wst:Status>
```

```
</wst:RequestSecurityTokenResponse>
```

$ИС_{sp}$ определя резултата от валидацията:

1) При валиден SAML токен

Елементът `/wst:RequestSecurityTokenResponse/wst:Status/wst:Code` съдържа следната стойност: `http://docs.oasis-open.org/ws-sx/ws-trust/200512/status/valid`

В този случай $ИС_{sp}$ продължава с изпълнението на заявката.

2) При невалиден SAML токен

Елементът `/wst:RequestSecurityTokenResponse/wst:Status/wst:Code` съдържа следната стойност: `http://docs.oasis-open.org/ws-sx/ws-trust/200512/status/invalid`

В този случай $ИС_{sp}$ спира изпълнението на заявката и обработва възникналото състояние според вътрешните правила на администрацията.

Следва списък с допълнителни компоненти, които фирма Бул Ес Ай ще реализира според техническото задание. Това са:

- Схема на обектните идентификатори в БеУ;
- Регистър на служителите в държавната администрация;
- Регистър на ресурсите в държавната администрация;
- Справочник за атрибути.

Следва описание на начина на реализация на компонентите

7.3 Схема на обектните идентификатори в БеУ

7.3.1 Описание

Обектен идентификатор (OID) е широко използван идентификационен механизъм, разработен съвместно от ITU-T и ISO / IEC за именуване на какъвто и да е вид обекти, концепция или "нещо" с глобално недвусмислено име. Концепцията се основава на йерархична структура на имена на базата на OID дърво.

Информационните системи в администрациите, използващи инфраструктурата на електронното управление, ще се дефинират като ресурси в специализирана система (виж 7.4 Регистър на ресурсите в Държавната Администрация). Неотменна характеристика на всеки ресурс е уникален идентификатор според приетата OID схема.

За да се избегне двусмислието при реферирането към ресурси в БеУ, всички заявки за електронни услуги, генерирани от информационни системи в администрациите, ще използват идентификаторите от предложената схема.

Примерна структура на OID схема е дадена долу:

1.2.100 ISO/Member-body/BG

1.2.100.**1**…BG e-Government

1.2.100.1.**1**…Administrations

1.2.100.1.1.**1**…Administration A

1.2.100.1.1.1.**1**...Information Systems

1.2.100.1.1.1.1.**1**...Information System A

1.2.100.1.1.1.1.1.**1**...Electronic Services

1.2.100.1.1.1.1.**1**.1.**1**...Electronic Service A1

1.2.100.1.1.1.1.1.1.**2**…Electronic Service A2

1.2.100.1.1.1.1.**1**...Information System B

1.2.100.1.1.1.1.1.**1**...Electronic Services

1.2.100.1.1.1.1.1.1.**1**...Electronic Service B1

1.2.100.1.1.1.1.1.1.**2**…Electronic Service B2

…

1.2.100.1.1.**2**...Administration B

...

1.2.100.2…BG e-Governance LDAP Elements

1.2.100.2.1…AttributeTypes

1.2.100.2.1.1…custom-Attribute

…

1.2.100.2.2…ObjectClasses

1.2.100.2.2.1…custom-ObjectClass

…

Схемата има за начало елемента с oid 1.2.100.1. Структурата на този идентификатор е следната:

1. – организация ISO (ISO)

2. – кодове на държавите-членки (Member-body)

100. – код на Република България (BG)

1. – клон на идентификаторите на БеУ (BG e-Government)

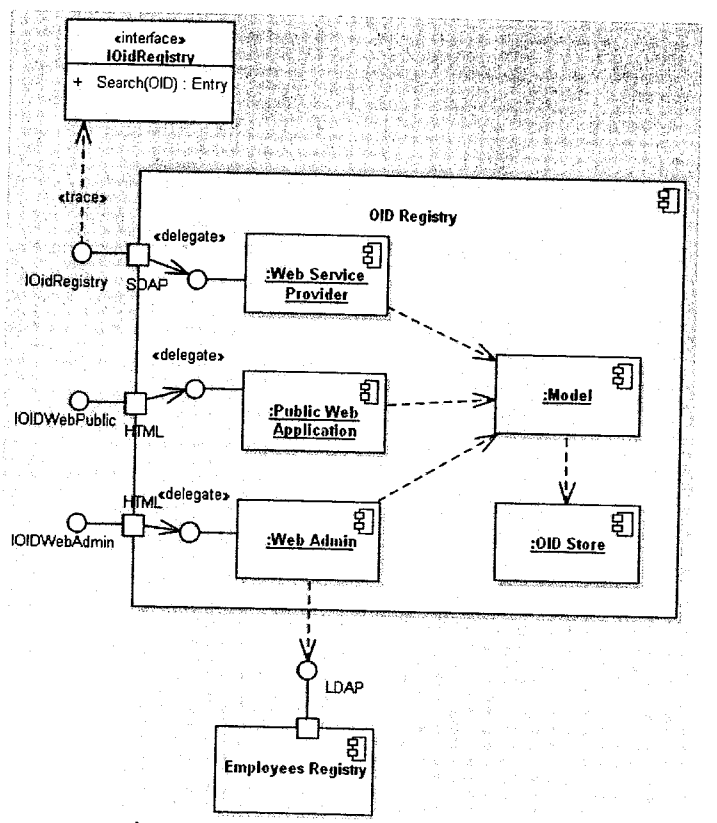
В схемата ще се поддържат обектни идентификатори в два основни под-клона на възел 1.2.100.1 (ISO/Member-body/BG/ BG e-Government):

- Администрации с техните информационни системи и предлагани електронни административни услуги
  Този клон би имал oid 1.2.100.1.1;
- Специфични за БеУ LDAP елементи (класове и атрибути) – това са елементи, които не са част от стандартната схема X.500/LDAP Attribute Profile, например като: „department" („отдел") с примерен код 1.2.100.2.1.1
  Този клон би имал oid 1.2.100.1.2.

7.3.2 Реализация

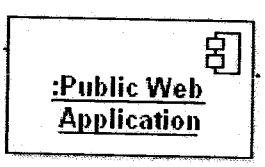
7.3.2.1 Компоненти

Регистърът на обектни идентификатори (РОИ) ще бъде изграден от следните компоненти:



Фигура 10 РОИ – Логическа архитектура

7.3.2.2 Уеб базирано приложение за публичен достъп



Ще позволява изобразяване на дървото с oid и търсене по: части от наименованието на администрация, части от наименованието на информационна система, по части от oid и др. Системата ще има само справочен характер и ще бъде публично достъпна през технологичния портал на БеУ.

Потребителският интерфейс на системата ще работи коректно с минимална разделителна способност 1024x768 и ще поддържа следните видове уеб браузъри и версии:

- Microsoft Internet Explorer 8 и по-висока;
- Firefox 24 и по-висока;
- Chrome 22 и по-висока.

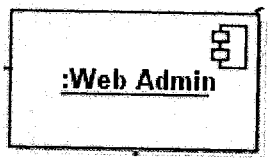
Ще бъде реализирано като JBoss SEAM приложение по технологията Facelet с JBoss RichFaces.

7.3.2.3 Уеб услуга за достъп до регистъра

Предоставя SOAP уеб услуга за търсене на идентификатори и извличане на наличната информация за тях. Ще позволява интеграция с други системи, например Порталът на БеУ ще може да изобразява данните в портлет при необходимост.

Ще бъде реализирана с технологията JAX-WS, предоставяна от JBoss WS – част от JBoss Application Server.

7.3.2.4 Уеб базирано приложение за поддържане на обектните идентификатори



Предоставя функционалност за създаване на нови и редактиране на съществуващи записи с обектни идентификатори. Приложението ще бъде достъпно само за специално оторизирани служители в ДА, които ще имат право да създават нови, и да редактират съществуващи записи в регистъра.

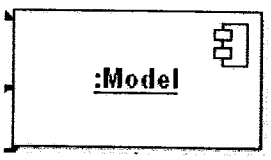
Потребителският интерфейс на системата ще работи коректно с минимална разделителна способност 1024x768 и ще поддържа следните видове уеб браузъри и версии:

- Microsoft Internet Explorer 8 и по-висока;
- Firefox 24 и по-висока;
- Chrome 22 и по-висока.

Автентикацията на потребителите ще се основава на „Регистър на служителите в Държавната Администрация" (7.5).

Ще бъде реализирано като JBoss SEAM приложение по технологията Facelet с JBoss RichFaces.

7.3.2.5 Бизнес логика



Реализира логиката за валидиране и манипулиране (създаване, промяна) на обектните идентификатори.

Ще бъде реализирана като Java EJB компоненти.

7.3.2.6 Хранилище за OID данни

Тук се съхранява йерархичната структура на обектните идентификатори. За всеки елемент ще се поддържа най-малко следната информация:

- Код на oid;
- Наименование и предназначение;
- Дата на създаване;
- Подробно описание;
- Отговорна администрация, която поддържа елемента.

Ще бъде реализирано като набор от таблици, изгледи, индекси и други артефакти в релационна СУБД.

7.4 Регистър на ресурсите в Държавната Администрация

7.4.1 Описание

Регистърът на ресурсите в Държавната Администрация (РРЕС) ще поддържа информация за информационните системи в администрациите и за услугите, които администрациите предоставят.

Компонентът ще предоставя следната основна функционалност:

- Каталогизиране, организиране и поддържане на ресурси;
- Автоматизиране на процеса за преглед и поддържане на ресурсите;
- Средства за анализ на влиянието на промените на един ресурс върху останалите ресурси, посредством свързаност на ресурсите.

За всеки един ресурс ще се поддържат най-малко следните характеристики:

- Наименование на ресурс;
- Подробно описание;
- Идентификатор на ресурс според OID схемата на е-Управление;
- Вид на ресурса;
- Собственик на ресурса – администрация, контакти, администратори и др.;

- Статус – според governance процес;
- Класификатори (ще позволяват групиране на ресурси по разнообразни критерии);
- Ръководство за използване на ресурса;
- Процес за управление на жизнения цикъл на ресурса;
- Свързани ресурси.

Поддържа и управлява каталог на бизнес и ИТ ресурси в домейна на електронното управление, като:

- Информационни системи;
- Уеб услуги (описание).

За всяка информационна система ще се поддържат най-малко следните характеристики:

- Наименование на информационната система;
- Обектен идентификатор (oid) на информационната система;
- Наименование на администрацията, собственик на системата;
- Обектен идентификатор на администрацията (oid);
- Допълнителна информация (разработчик, операционна система, сървър за приложения и др.);
- Услуги, които информационната система предоставя.

За всяка услуга ще да се поддържат най-малко следните характеристики:

- Наименование на услугата;
- Обектен идентификатор (oid) на услугата;
- Версия;
- URL на SOAP порт на услугата;
- Допълнителна информация (изисквания за сигурност, особености в реализацията и др.).

Информацията в регистъра ще се съхранява в различни хранилища: в LDAP регистър (йерархична структура на администрациите, информационните системи и бизнес услугите), в UDDI (информация за уеб услуги) и в релационна база данни (допълнителна консолидираща информация).

Регистърът ще играе ролята на обединяващо звено, където информацията от различните хранилища ще се консолидира и управлява.

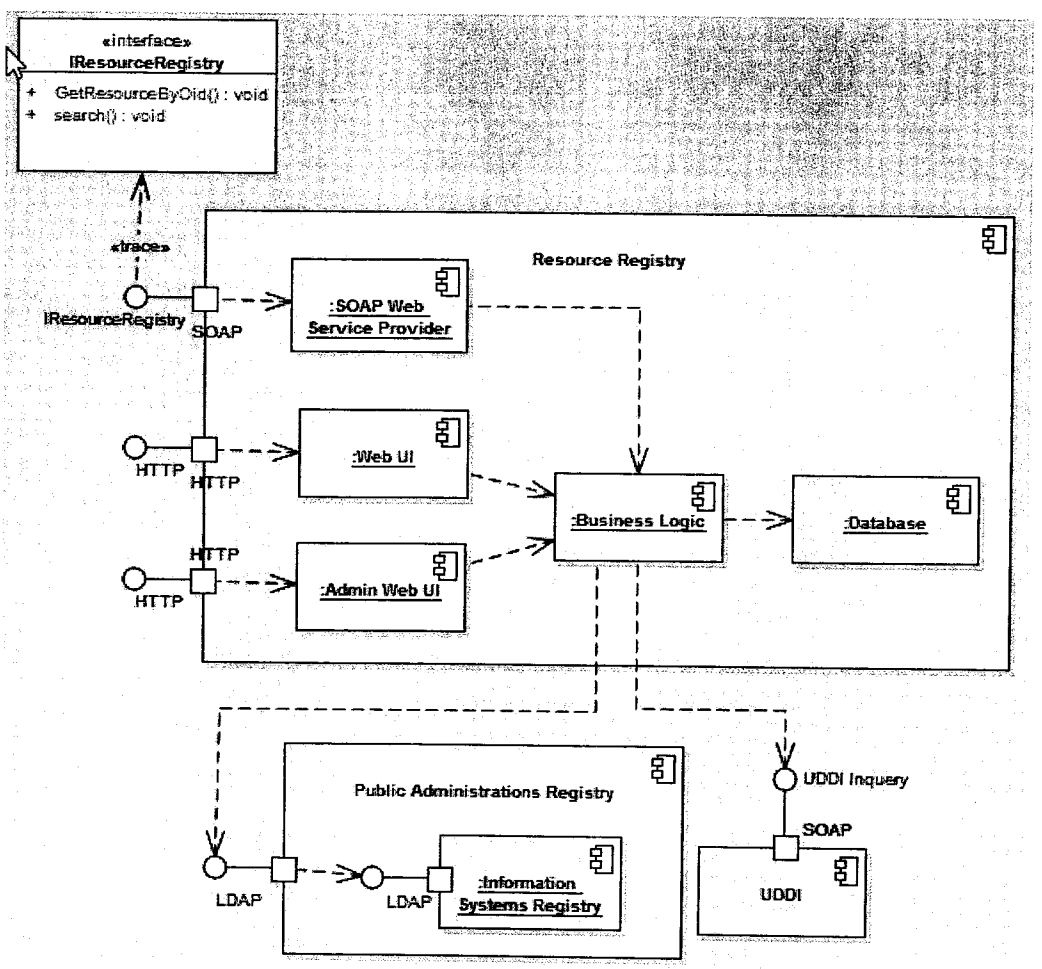
Системата ще извлича информация за уеб услугите от инсталирания в МТИТС UDDI регистър на услуги.

РРЕС ще реализира процеси за управление на жизнения цикъл на различните видове ресурси: създаване, одобрение, публикуване, деактивиране, премахване. Управлението на жизнения цикъл ще става с помощта на приложение. Приложението трябва да реализира и базова функционалност за търсене на ресурси по: вид, обектен идентификатор, части от наименованието (на информационна система или услуга).

Администратор на централно ниво ще може да дефинира през приложението публично достъпните ресурси в домейна на БеУ.

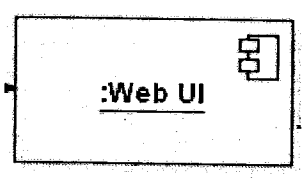
7.4.2 Реализация

7.4.2.1 Компоненти



Фигура 11 РРЕС – Логическа архитектура

7.4.2.1.1 Уеб базирано приложение за публичен достъп



Публично достъпен през технологичния портал на БеУ. Ще позволява изобразяване на ресурсите и търсене по: части от наименованието на администрация, по вид ресурс, части от наименованието на ресурс, по части от oid и др.

Системата ще има само справочен характер.

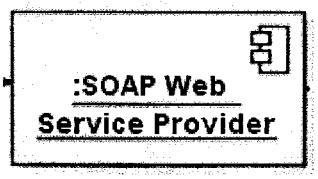
Приложението ще реализира функционалност за търсене на ресурси по: вид, обектен идентификатор, части от наименованието (на информационна система или услуга).

Потребителският интерфейс на системата ще работи коректно с минимална разделителна способност 1024x768 и ще поддържа следните видове уеб браузъри и версии:

- Microsoft Internet Explorer 8 и по-висока;
- Firefox 24 и по-висока;
- Chrome 22 и по-висока.

Ще бъде реализирано като JBoss SEAM приложение по технологията Facelet с JBoss RichFaces.

7.4.2.1.2 Уеб услуга за достъп до регистъра



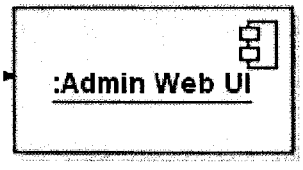
РРЕС ще предостави интерфейс за достъп за други информационни системи. По подаден списък от обектни идентификатори (oid) РРЕС трябва да извлича описание на съответстващите информационни система и/или услуги.

Този интерфейс ще бъде реализиран като SOAP уеб услуга. Уеб услугата ще реализира функционалността за извличане на описание на информационни система и/или услуги по подаден списък с обектни идентификатори.

Такава система например може да е Порталът на БеУ, където данни от регистъра при необходимост ще могат да бъдат изобразявани в портлети.

Ще бъде реализирана с технологията JAX-WS, предоставяна от JBoss WS – част от JBoss Application Server.

7.4.2.1.3 Уеб базирано приложение за поддържане на ресурсите



Ще бъде достъпен само за оторизирани служители в ДА, които ще имат право да създават нови и да редактират съществуващи записи в регистъра, както и да променят статуса на ресурс (според процеса за управление на жизнения цикъл).

Идентификацията на потребителите ще става с потребителско име и парола. Правата за достъп ще се определят от роли.

Автентикация на потребителите ще се основава на Регистър на служителите в Държавната (виж 7.5).

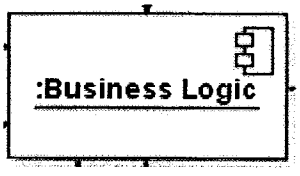
При промяна на запис системата ще генерира събитие в Системата за обработка на бизнес събития (виж т. 10), така че други системи да бъдат известявани при промени в регистъра.

Потребителският интерфейс на системата ще работи коректно с минимална разделителна способност 1024x768 и ще поддържа следните видове уеб браузъри и версии:

- Microsoft Internet Explorer 8 и по-висока;
- Firefox 24 и по-висока;
- Chrome 22 и по-висока.

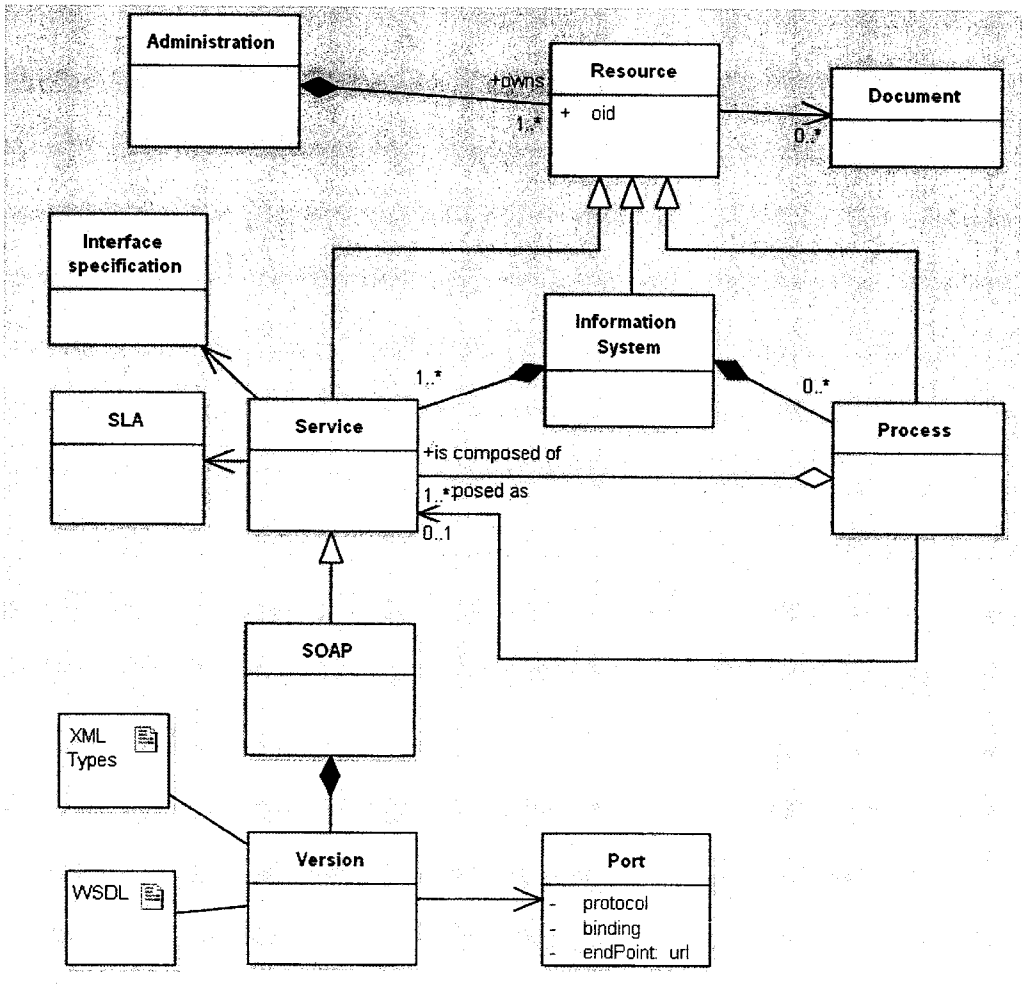
Ще бъде реализирано като JBoss SEAM приложение по технологията Facelet с JBoss RichFaces.

7.4.2.1.4 Бизнес логика



Компонентът ще реализира процеси за управление на жизнения цикъл на различните видове ресурси: създаване, одобрение, публикуване, деактивиране, премахване. Процесите ще бъдат описвани като BPMN 2 процеси. Интерпретирането на тези процеси ще се осъществява от система за управление на бизнес процеси Activity BPM. Това ще позволи без намеса в кода на системата лесна промяна на съществуващите процеси и добавяне на нови процеси при дефиниране на нови видове ресурси.

Данните, които се ще се поддържат от регистъра, са показани в следващата фигура:



Фигура 12 Видове ресурси, поддържани от компонента

За всяка администрация компонентът ще поддържа ресурси. Всеки ресурс има уникален обектен идентификатор (oid) според схемата на обектните идентификатори в БеУ.

За всеки ресурс има един или повече документа, които описват ресурса.

Информационните системи, които предоставят електронни административни услуги, са ресурс по смисъла на БеУ.

Една информационна система може да предоставя няколко услуги и бизнес процеси.

Входната точка за бизнес процес може да е уеб услуга.

Всяка SOAP уеб услуга има една или повече версии.

Всяка версия на услугата дефинира порт с протокол, адрес и свързване (binding).

Всяка версия има описание на данните (XML схема) и описание на услугата (WSDL).

Жизненият цикъл на всеки ресурс се управлява от governance процес.

Примерен governance процес „Създаване на ресурс Бизнес услуга" ще се състои от следните стъпки:

1. Определяне на необходимост от създаване на нова електронна услуга
2. Осигуряване на финансиране за разработка на услугата
3. Анализ на административните процеси, които услугата ще автоматизира
4. Разбиване на формални стъпки (уеб услуги)
5. Анализ на изискванията
6. Реализация на услугата
7. Тестване на услугата
8. Публикуване

Приложението ще предостави функционалност за:

- Избор на администрация и информационна система
- Заявяване на oid за новата услуга
- Създаване на нов запис в регистъра от вид „SOAP уеб услуга" – попълване на описание, наименование, качване на wsdl, попълване на специфични данни за UDDI (класификатори: oid на администрация, oid на информационна система, oid на версия на уеб услугата – SOAP порт).

При запис системата ще:

- създаде запис в локалната база данни с допълнителна информация: спецификация на услугата,
- създаде с LDAP заявка запис в LDAP регистъра
- създаде с UDDI заявка запис в Регистъра с услуги UDDI запис за уеб услугата .За всяка администрация в UDDI ще има създадено businessEntity. В това businessEntity системата ще създаде запис от вид businessService и ще регистрира съответните характеристики в bindingTemplate: url на wsdl, end point и в набор от tModel-и съответните класификатори (oid на администрация, oid на информационна система, oid на версия на уеб услугата – SOAP порт).

Системата ще промени статуса на уеб услугата на „Готова за публикуване”

Определен брой отговорни служители ще валидират данните и при одобрение на всички от тях услугата ще бъде достъпна за използване.

Системата ще извести всички заинтересувани посредством публикуване на събитие в

След което започва да тече процесът по поддръжка.

Бизнес логиката ще бъде реализирана като Java EJB компоненти.

Управлението на governance процесите ще става с продукта Activiti BPM.

7.5 Регистър на служителите в Държавната Администрация

7.5.1 Описание

Регистърът на служителите в Държавната администрация (РСлДА) ще поддържа актуален списък на всички служители във всички администрации, които участват в предоставянето на електронни услуги в БеУ.

Регистърът ще предоставя справочни услуги за преглед на държавните органи и техните сфери на отговорност. Тези услуги ще предоставят необходимата информация за администрациите (адрес, контакти, служители, длъжности и др.). Информацията ще може да бъде извличана със стандартното търсене (например с LDAP заявки).

Директорията ще позволява прозрачно търсене и локализиране на отговорните администрации и служители.

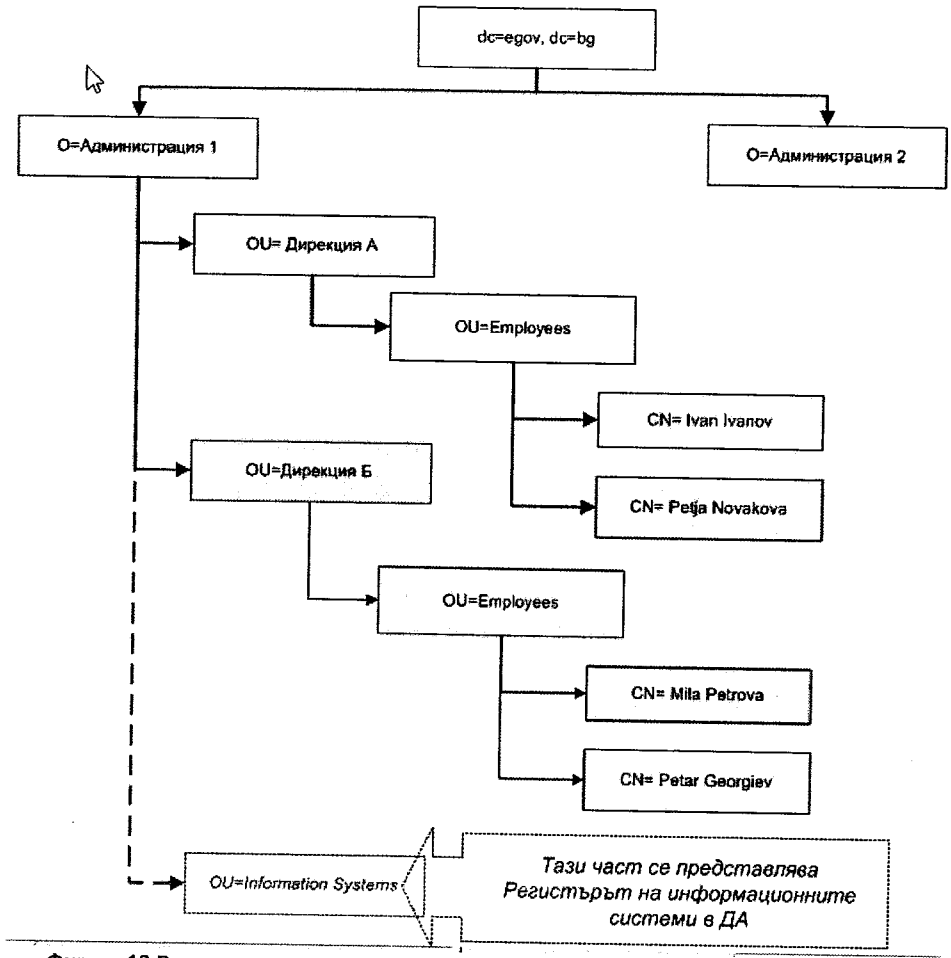
Услугите ще се използват от различни вътрешни и външни информационни системи и ще изпълняват основно две функции:

- първата е реализация на директория, представляваща всички администрации в Република България. Директорията ще съдържа данни за адреси и телефонен указател на публичните власти. Директорията ще поддържа пълно текстово търсене в йерархичната структура на администрациите;
- втората функция е за нуждите на процеса на идентификация, автетикация и оторизация на служителите в администрациите. Системата за електронна авторизация (еАвт) през Справочника за атрибути ще изисква информация на основата на присъстващия в SAML токен идентификатор на физическо лице (ЕГН/ЛНЧ). Администрациите ще се грижат да поддържат актуален списък с потребители, техните позиции и отговорни роли.

За реализация на справочни услуги ще се използва LDAP регистър. За лесна класификация в LDAP регистъра на обектите и техните характеристики ще се разработи специална OID схема.

Регистърът ще поддържа дървовидна структура на администрациите. Ще бъде разработена специална LDAP схема, дефинираща нови класове и атрибути в допълнение на стандартната LDAP схема. С помощта на тези класове и атрибути ще бъде поддържана информация за информационните системи в администрациите, както

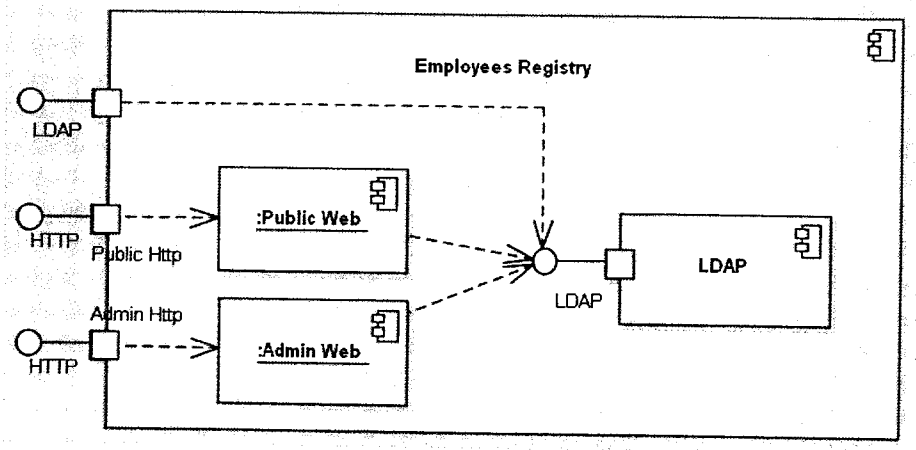
и за услугите, които информационните системи предоставят. Пример за структура на Регистъра.



Фигура 13 Регистър на служителите в ДА – логическа структура на информацията

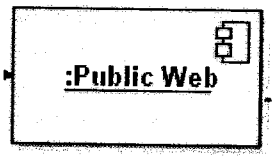
Забележка: Изобразената на Фигура 13 структура на информацията е само с илюстративен характер за целите на техническото предложение. Действителната структура може да се различава с цел оптимизиране на търсенето, избягване на повторения и др.

7.5.1.1 Реализация



Фигура 14 Регистър на служителите в ДА – Логическа архитектура

7.5.1.1.1 Уеб базирано приложение за публичен достъп



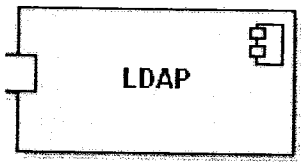
Приложението ще предостави функционалност за търсене на служители в регистъра по части от наименование на администрация, отдел, имена на служителя, други негови характеристики. Приложението ще е публично достъпно през технологичния портал на БеУ. Системата ще има само справочен характер.

Потребителският интерфейс на системата ще работи коректно с минимална разделителна способност 1024x768 и ще поддържа следните видове уеб браузъри и версии:

- Microsoft Internet Explorer 8 и по-висока;
- Firefox 24 и по-висока;
- Chrome 22 и по-висока.

Ще бъде реализирано като JBoss SEAM приложение по технологията Facelet с JBoss RichFaces.

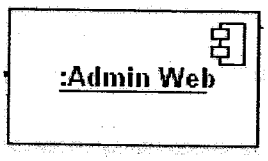
7.5.1.1.2 LDAP регистър



Реализацията на справочни услуги предполага наличието на актуална директория. Такава директория е LDAP.

За реализация ще се използва продуктът с отворен код OpenLdap. Поддържането на запис в регистъра ще се извършва с предоставената от OpenLdap административна конзола.

7.5.1.1.3 Уеб базирано приложение за поддържане на записите в регистъра



Ще бъде достъпен само за специално оторизирани служители в ДА, които ще имат право да създават нови, и да редактират съществуващи записи в LDAP регистъра.

Потребителският интерфейс на системата ще работи коректно с

минимална разделителна способност 1024x768 и ще поддържа следните видове уеб браузъри и версии:

- Microsoft Internet Explorer 8 и по-висока;
- Firefox 24 и по-висока;
- Chrome 22 и по-висока.

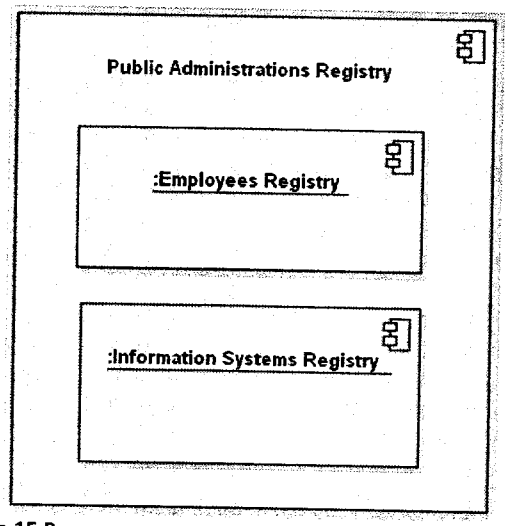
Ще бъде реализирано като JBoss SEAM приложение по технологията Facelet с JBoss RichFaces.

7.6 Регистър на администрациите

Регистърът на администрациите в БеУ обединява два основни регистъра:

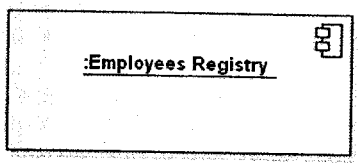
- Регистър на служителите в ДА;
- Регистър на информационните системи в ДА.

Логическата структура на регистъра е изобразена на следващата фигура:



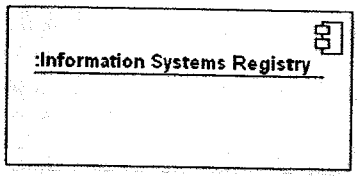
Фигура 15 Регистър на администрациите в ДА – компоненти

7.6.1 Регистър на служителите



Описан е подробно в „Регистър на служителите в Държавната Администрация" (7.5)

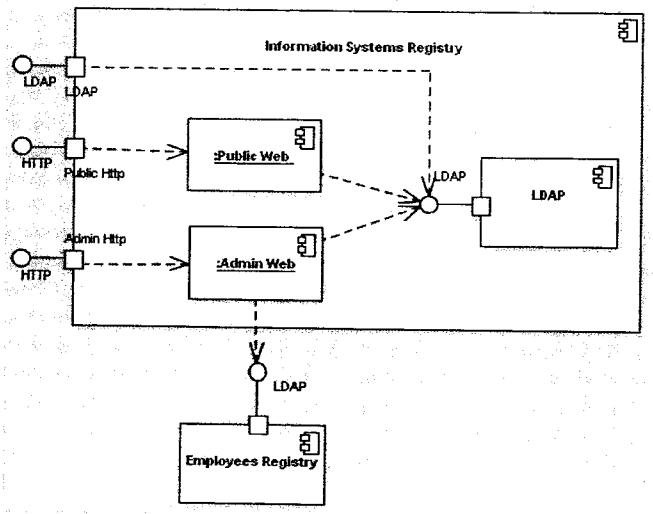
7.6.2 Регистър на информационните системи



Регистърът на информационните системи поддържа данни за информационните системи в държавните администрации и услугите, които тези системи предоставят.

Структурата на поддържаните данни е йерархична.

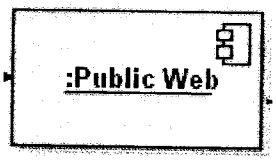
Логическа структура на регистъра е изобразена на следващата фигура:



Фигура 16 Регистър на информационните системи в ДА – компоненти

7.6.2.1 Компоненти

7.6.2.1.1 Уеб базирано приложение за публичен достъп



Приложението ще предостави функционалност за търсене в регистъра по части от наименование на администрация, на предоставяна услуга и по местонахождение на Публично достъпен през технологичния портал на БеУ. Ще позволява изобразяване на дървото с oid и търсене по: части от наименованието на администрация, части от наименованието на информационна система, по части от oid и др.

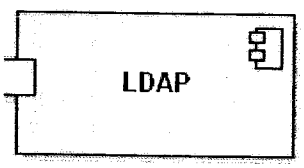
Системата ще има само справочен характер.

Потребителският интерфейс на системата ще работи коректно с минимална разделителна способност 1024x768 и ще поддържа следните видове уеб браузъри и версии:

- Microsoft Internet Explorer 8 и по-висока;
- Firefox 24 и по-висока;
- Chrome 22 и по-висока.

Ще бъде реализирано като JBoss SEAM приложение по технологията Facelet с JBoss RichFaces.

7.6.2.1.2 LDAP регистър



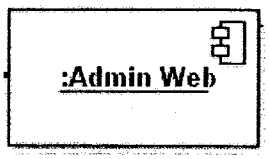
LDAP Регистърът е стандартизиран компонент, който ще включва:

Реализацията на справочни услуги предполага наличието на актуална директория. Поддържането на такава директория

изисква пълно сътрудничеството на всички държавни органи, които трябва да предоставят актуални данни за промените в състава и структурата си на регулярна основа с LDIF.

За реализация ще се използва продуктът с отворен код OpenLdap. Поддържането на запис в регистъра ще се извършва с предоставената от OpenLdap административна конзола.

7.6.2.1.3 Уеб базирано приложение за поддържане на записите в регистъра



Ще бъде достъпен само за специално оторизирани служители в ДА, които ще имат право да създават нови, и да редактират съществуващи записи в регистъра.

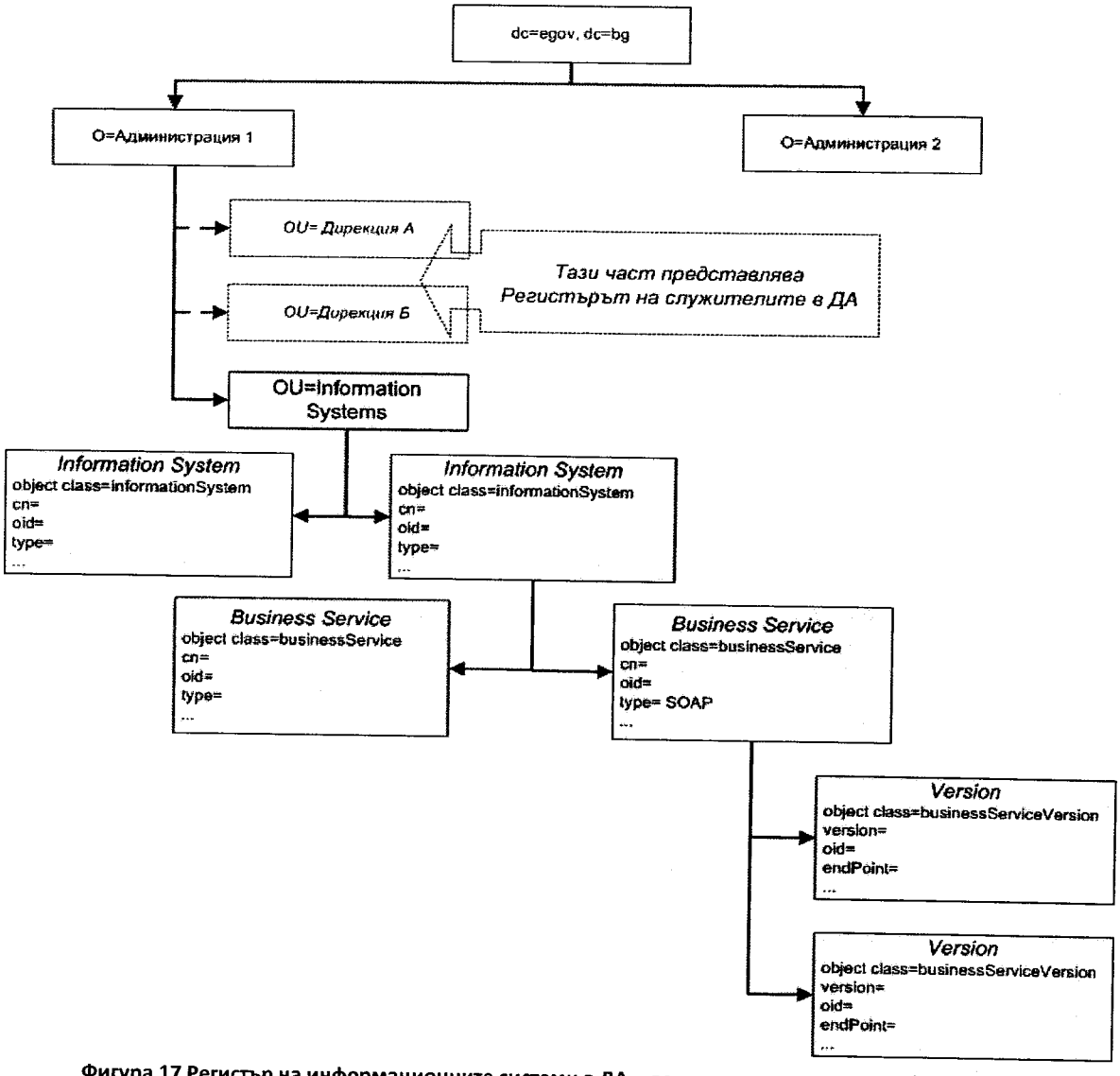
Автентикация ще се основава на Регистъра на служителите в ДА по протокол LDAP.

Потребителският интерфейс на системата ще работи коректно с минимална разделителна способност 1024x768 и ще поддържа следните видове уеб браузъри и версии:

- Microsoft Internet Explorer 8 и по-висока;
- Firefox 24 и по-висока;
- Chrome 22 и по-висока.

Ще бъде реализирано като JBoss SEAM приложение по технологията Facelet с JBoss RichFaces.

Данните за ресурсите, които се поддържат в регистъра, са изобразени на следващата диаграма:



Фигура 17 Регистър на информационните системи в ДА – логическа структура на информацията

Забележка: Изобразената на Фигура 17 структура на информацията е само с илюстративен характер за целите на техническото предложение. Действителната структура може да се различава с цел оптимизиране на търсенето, избягване на повторения и др.

Всяка администрация има една или няколко информационни системи, които предоставят бизнес услуги.

Всяка бизнес услуга има: наименование, обектен идентификатор (oid) вид (SOAP, REST).

Всяка бизнес услуга има една или повече версии.

Всяка версия има: идентификатор на версията, обектен идентификатор (oid), краен адрес (endPoint).

За нуждите на БеУ фирма Бул Ес Ай ще разработи допълнителна LDAP схема с нови класове и атрибути, например:

- за обект „Информационна система" ще бъде създаден нов клас (object class = informationSystem) с допълнителни ldap атрибути;
- за обект „Административна услуга" ще бъде създаден нов клас (object class = businessService) с допълнителни ldap атрибути;
- за обект „Версия на административна услуга" ще бъде създаден нов клас (object class = businessServiceVersion) с допълнителни ldap атрибути.

Информация за допълнителната LDAP схема ще бъде публикувана в технологичния портал на БеУ и ще публично достъпна за всички разработчици на електронни административни услуги.

7.7 Справочник за атрибути

7.7.1 Описание

Компонентът „Справочник за атрибути" (САтр) ще реализира функционалност за предоставяне на допълнителни характеристики (атрибути) за физически лица, които липсват в съдържанието на идентифициращия SAML токен. Такива атрибути могат да бъдат: заемана длъжност в администрацията, адрес на електронна поща и др. Характеристиките ще се поддържат в LDAP в Регистъра на служителите в ДА (виж 7.5).

САтр ще предостави функционалност за управление на жизнения цикъл на атрибутите: създаване, публикуване, редактиране и изтриване.

САтр ще предостави функционалност за извличане на атрибути по зададени критерии за търсене.

Атрибутите ще бъдат класифицирани според oid схемата на БеУ (виж 7.3 Схема на обектните идентификатори в БеУ).

Заявката за извличане ще има следния формализиран вид:

R(Subject id, Attribute 1 oid, Attribute 2 oid, …., Attribute N oid), където:

Subject id е идентификатор на субект (секторен псевдоним или други)

Attribute 1..N oid – обектен идентификатор на атрибут

Резултатът от изпълнението е:

A(Subject id, Attribute 1 value [], Attribute 2 value[], …., Attribute N value[]), където:

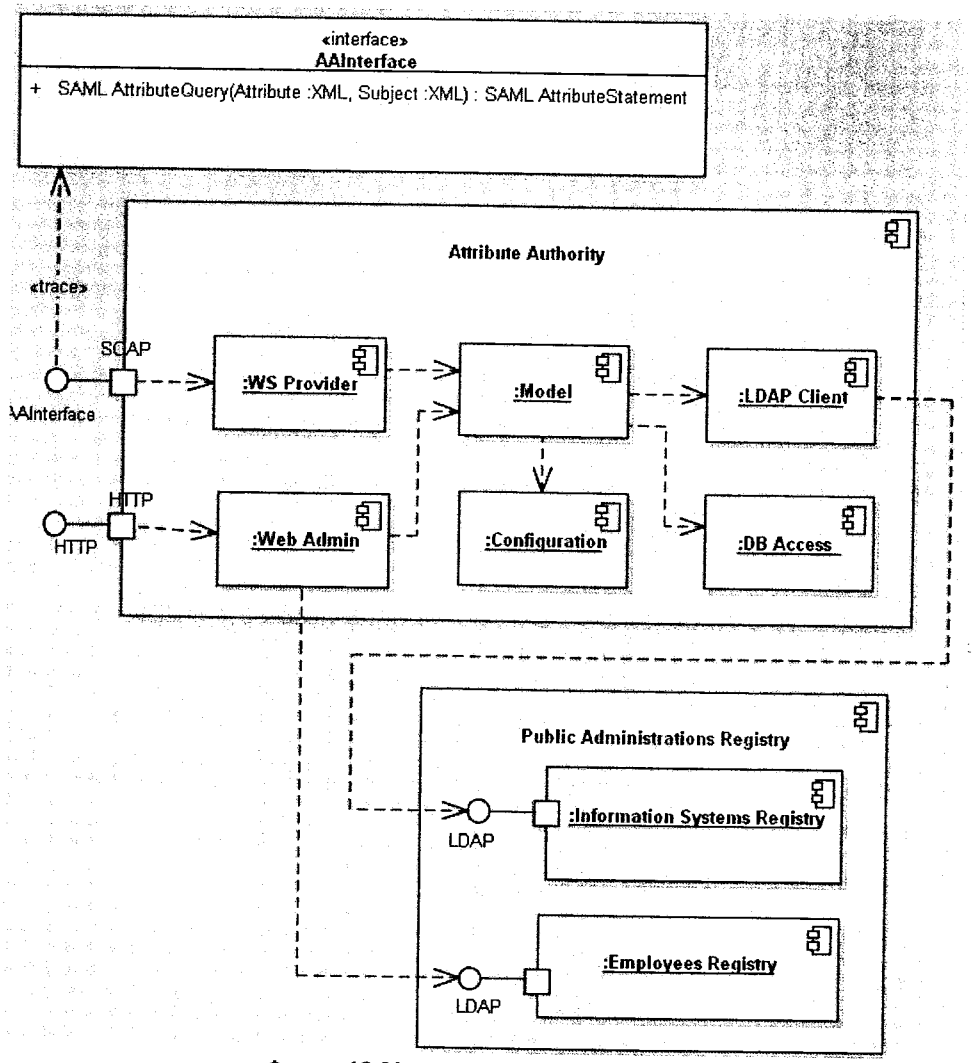
Subject id е идентификатор на субект, за който се извличат атрибутите

Attribute 1..N value [] е списък със стойности на съответния атрибут.

7.7.2 Реализация

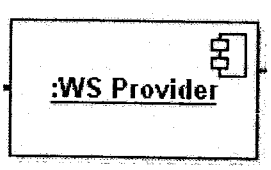
7.7.2.1 Компоненти

Логическата структура на компонента е изобразена на следващата фигура:



Фигура 18 САтр – Логическа архитектура

7.7.2.1.1 Уеб услуга за достъп до справочника



Компонентът ще предостави уеб услуга (SOAP/HTTPS) за достъп до справочника от други информационни системи.

SOAP уеб услуга ще реализира стандартна заявка SAML Attribute Query. Пример за такава заявка е даден по-долу:

```
<soap-env:Envelope xmlns:soap-env:="http://schemas.xmlsoap.org/soap/envelope/">
  <soap-env:Header/>
```

```
<soap-env:Body>

    <samlp:Request
        xmlns:saml="urn:oasis:names:tc:SAML:2.0:assertion"
        xmlns:samlp="urn:oasis:names:tc:SAML:2.0:protocol"
        MajorVersion="1" MinorVersion="1"
        IssueInstant="2004-12-05T09:22:04Z"
        RequestID="...">

        <samlp:AttributeQuery
            xmlns:saml="urn:oasis:names:tc:SAML:2.0:assertion"
            xmlns:samlp="urn:oasis:names:tc:SAML:2.0:protocol"
            ID="..."
            Version="2.0"
            IssueInstant="2006-07-17T20:31:40">

    <saml:Issuer
      Format="urn:oasis:names:tc:SAML:1.1:nameid-format:X509SubjectName">
      ...
    </saml:Issuer>

   <saml:Subject>
      <saml:NameID Format="urn:oasis:names:tc:SAML:2.0:assertion#X509SubjectName">
            CN=Иван Иванов,OU=Дирекция,O=МТИТС,CN=dir,DC=egov,DC=bg
        </saml:NameID>
    </saml:Subject>

<saml:Attribute
      NameFormat="urn:oasis:names:tc:SAML:2.0:assertion#X509SubjectName"
      Name="urn:oid:1.2.100.2.1.1"
      FriendlyName="department" />

<saml:Attribute
      NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:uri"
      Name="urn:oid:2.5.4.42"
      FriendlyName="givenName" />

   <saml:Attribute
      NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:uri"
      Name="urn:oid:1.3.6.1.4.1.1466.115.121.1.26"
      FriendlyName="mail" />

</samlp:AttributeQuery>
</samlp:Request>

</soap-env:Body>
</soap-env:Envelope>
```

Описанието на данните, които СпрАтр трябва да извлече, са описани в елементите saml:Attribute. В горния случай това са:

- Име → „givenName" (urn:oid:2.5.4.42)
- Отдел → „department" (urn:oid: 1.2.100.2.1.1)
- Адрес на електронна поща → „mail" (urn:oid:1.3.6.1.4.1.1466.115.121.1.26)

СпрА поддържа връзката между вид на атрибута, зададен с неговия обектен идентификатор (например urn:oid:2.5.4.42) и информационна система, която е отговорна за поддържането на информацията за съответния атрибут. Информационните системи са собственост на държавни агенции –администратори на първични данни

Компонентът СпрА ще позволява дефинирането на 2 вида интерфейси:

- LDAP;
- База данни.

Тъй като атрибутът `oid:2.5.4.42` е дефиниран в LDAP схема, съответната информационна система, която е отговорна за поддържането му, е компонентът „Регистър на служителите в Държавната Администрация" (виж 7.5).

СпрА генерира LDAP заявка към „Регистъра на служителите в ДА" (виж 7.5) по данните е елемента `saml:NameIdentifier` (това е `CN=Иван Иванов,OU=Дирекция,O=МТИТС,DC=dir,DC=egov,DC=bg`) и извлича съответните атрибути.

Регистърът на служителите в ДА извлича данните за служителя.

СпрА генерира отговор на заявката AttributeQuery. Отговорът има следното примерно съдържание:

```
<saml:Assertion
    xmlns:saml="urn:oasis:names:tc:SAML:2.0:assertion"
    …
    ID="..."
    Version="2.0"
    IssueInstant="2014-07-17T20:31:41">
    <saml:Issuer>...</saml:Issuer>
    <ds:Signature>...</ds:Signature>

    <saml:Subject>
      <saml:NameID
        Format="urn:oasis:names:tc:SAML:1.1:nameid-format:X509SubjectName">
        CN=Иван Иванов,OU=Дирекция,O=МТИТС,DC=dir,DC=egov,DC=bg
      </saml:NameID>
      ...
    </saml:Subject>
…
```

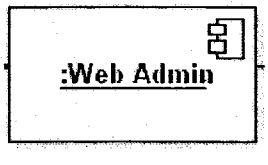
```
    <saml:AttributeStatement>
      <saml:Attribute
        xmlns:x500="urn:oasis:names:tc:SAML:2.0:profiles:attribute:X500"
        x500:Encoding="LDAP"
        NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:uri"
        Name="urn:oid:2.5.4.42"
        FriendlyName="givenName">
        <saml:AttributeValue
          xsi:type="xs:string">Ivan</saml:AttributeValue>
      </saml:Attribute>

      <saml:Attribute
        xmlns:x500="urn:oasis:names:tc:SAML:2.0:profiles:attribute:X500"
        x500:Encoding="LDAP"
        NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:uri"
        Name="urn:oid:2.5.4.4"
        FriendlyName="sureName">
        <saml:AttributeValue
          xsi:type="xs:string">Ivanov</saml:AttributeValue>
      </saml:Attribute>

      <saml:Attribute
        xmlns:x500="urn:oasis:names:tc:SAML:2.0:profiles:attribute:X500"
        x500:Encoding="LDAP"
        NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:uri"
        Name="urn:oid:1.3.6.1.4.1.1466.115.121.1.26"
        FriendlyName="mail">
        <saml:AttributeValue
          xsi:type="xs:string">ivan.ivanov@mtits.bg</saml:AttributeValue>
      </saml:Attribute>

    </saml:AttributeStatement>
  </saml:Assertion>
```

САтр генерира следния примерен отговор:

```
</soap-env:Envelope
  xmlns:soap-env:"http://schemas.xmlsoap.org/soap/envelope/";>
  <soap-env:Header/>
  </soap-env:Body>

    <samlp:Response
      xmlns:samlp="urn:oasis:names:tc:SAML:2.0:protocol"
      InResponseTo="aaf23196-1773-2113-474a-fe114412ab72"
      IssueInstant="2004-12-05T09:22:05Z"
      MajorVersion="1" MinorVersion="1"
      ResponseID="b07b804c-7c29-ea16-7300-4f3d6f7928ac">
      <samlp:Status>
        <samlp:StatusCode Value="samlp:Success"/>
      </samlp:Status>
      <saml:Assertion
```

```
        xmlns:saml="urn:oasis:names:tc:SAML:2.0:assertion"
        xmlns:xsd="http://www.w3.org/2001/XMLSchema";
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance";
        MajorVersion="1" MinorVersion="1"
        AssertionID="a144e8f3-adad-594a-9649-924517abe933"
        IssueInstant="2004-12-05T09:22:05Z"
        Issuer="http://bg.egov/idp">  <!-- IdP providerId -->
        <saml:Conditions
          NotBefore="2004-12-05T09:17:05Z"
          NotOnOrAfter="2004-12-05T09:52:05Z">
          <saml:AudienceRestrictionCondition>

            <saml:Audience>https://globus.org/gridshib</saml:Audience>
          </saml:AudienceRestrictionCondition>
        </saml:Conditions>

        <saml:AttributeStatement>
           <saml:Subject>
               <saml:NameID
                   Format="urn:oasis:names:tc:SAML:2.0:assertion#X509SubjectName">
                       ssPIN
               </saml:NameID>
            </saml:Subject>

             <saml:Attribute
                NameFormat="urn:oasis:names:tc:SAML:2.0:assertion#X509SubjectName"
                Name="2.5.4.11"
                FriendlyName="department">
                   OU=Finance Dept./O=MTITS/C=BG
             </saml:Attribute>
        </saml:AttributeStatement>
      </saml:Assertion>
    </samlp:Response>

  </soap-env:Body>
</soap-env:Envelope>
```

7.7.2.1.2 Уеб приложение за управление на справочника



Приложението за управление на справочника ще е уеб базирано. Ще бъде достъпно само за специално оторизирани служители в ДА, които ще имат право да създават нови, и да редактират съществуващи конфигурации.

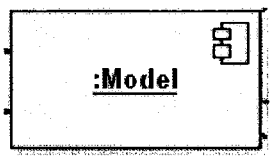
Идентификацията на потребителите ще да става с потребителско име и парола и ще се основава на Регистъра на служителите в ДА по протокол LDAP. Правата за достъп се определят от роли.

Потребителският интерфейс на системата трябва да работи коректно с минимална разделителна способност 1024x768 базирано и да поддържа следните видове уеб браузъри и версии:

- Microsoft Internet Explorer 8 и по-висока;
- Firefox 24 и по-висока;
- Chrome 22 и по-висока.

Ще бъде реализирано като JBoss SEAM приложение по технологията Facelet с JBoss RichFaces.

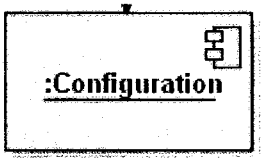
7.7.2.1.3 Бизнес логика



Реализира логиката за управление и извличане на конфигурации.

Бизнес логиката ще бъде реализирана като Java EJB компоненти.

7.7.2.1.4 Конфигурация



Конфигурацията определя видовете източници на атрибути.

САтр ще може да достъпва следните видове източници:

- LDAP;
- Бази данни.

Конфигурацията ще поддържа връзката между:

Обектен идентификатор (oid) на атрибут и източник (ИС, която съхранява информация за атрибута)

Първоначално единствената ИС, която ще предоставя атрибути, е Регистърът на служителите в ДА.

Фирма Бул Ес Ай ООД ще разработи системата по такъв начин, че в бъдеще да могат да бъдат включвани и регистрите на администраторите на първични данни, като източници на атрибути.

7.7.2.1.4.1 Източник LDAP

Основни параметри на конфигурацията ще бъдат:

- Адрес на LDAP сървър: IP адрес на хост и порт;
- Base DN;

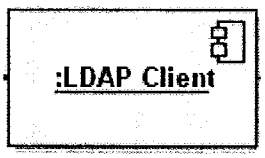
- Данни за профил в LDAP регистъра, необходими за осъществяване на връзка;
- LDAP шаблон на заявка

7.7.2.1.4.2 Източник БД

Основни параметри на конфигурацията ще бъдат:

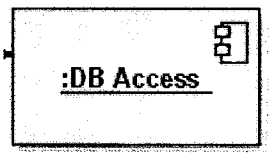
- Адрес на сървър за бази данни: IP адрес на хост и порт;
- Данни за базата данни
- Данни за jdbc драйвер;
- Данни за профил в СУБД регистъра, необходими за осъществяване на връзка;
- Шаблон на SQL заявка.

7.7.2.1.4.3 Компонент за връзка с LDAP регистри



Предназначениеро на този компонент е да осъществи връзка с LDAP регистър – източник на информация за атрибути по протокол LDAP според данните в конфигурацията. Първоначално такъв източник на данни ще е само компонентът „Регистър на служителите в Държавната Администрация" (7.5).

7.7.2.1.4.4 Компонент за връзка с Релационни СУБД



Предназначението на този компонент е да осъществи връзка с източници на информация за атрибути (Релационни СУБД) по протокол JDBC според данните в конфигурацията.

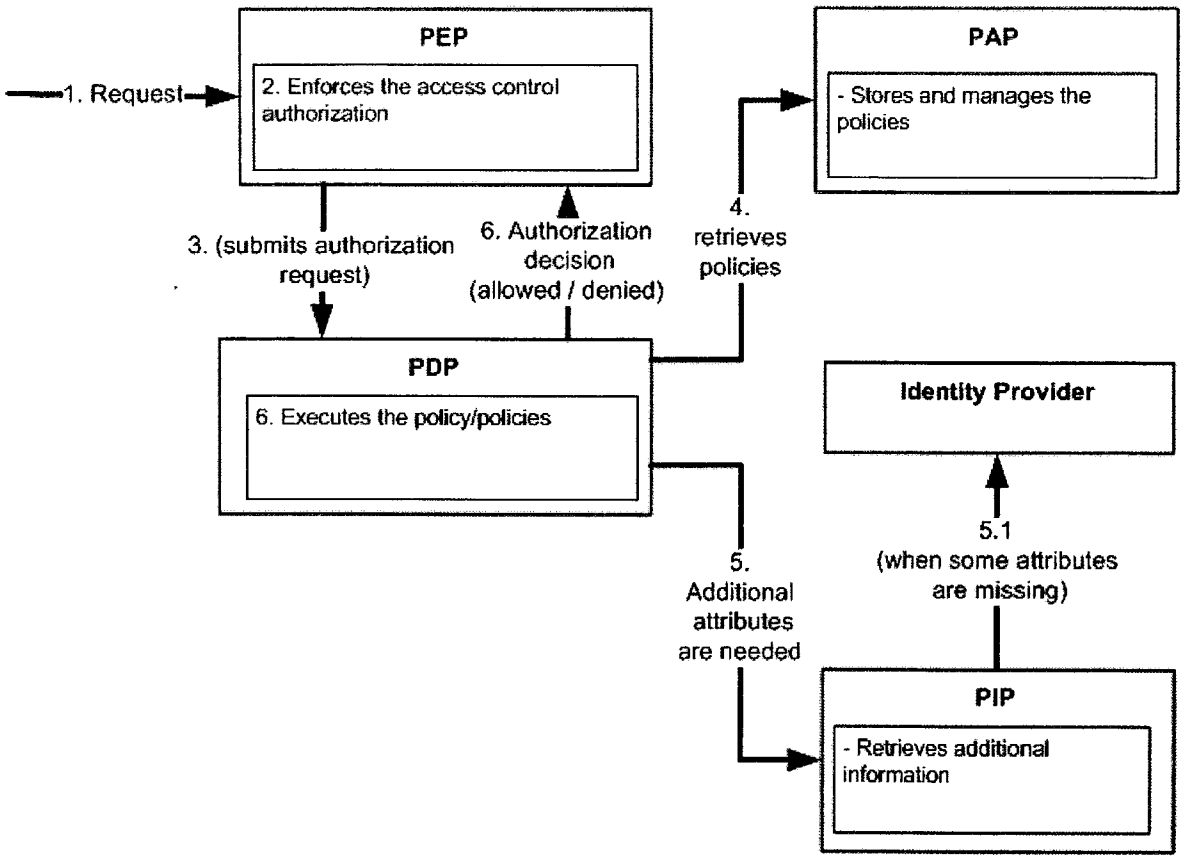
8 МЕТОДИКА ЗА РЕАЛИЗАЦИЯ НА КОМПОНЕНТ ЗА ЕЛЕКТРОННА ОТОРИЗАЦИЯ, ПОЗВОЛЯВАЩ ДЕФИНИРАНЕ НА ГЪВКАВИ ПРАВИЛА ЗА РАЗРЕШАВАНЕ ИЛИ ОГРАНИЧАВАНЕ НА ДОСТЪПА ДО СИСТЕМНИ РЕСУРСИ

8.1 Описание

Развитието на средата на БеУ с добавяне на нови услуги, системи и компоненти налага разработването на централизирана система за управление на достъпа до системните ресурси в БеУ. Ресурси, достъпът до които е ограничен и изисква оторизация, са например: информационните системи в администрациите, уеб услуги, интерфейси, видове данни и др.

Изграждането на Компонент за електронна оторизация (еОтор) ще способства да се въведат по-строги политики и единен контрол на достъпа до ресурсите в администрациите. Администрациите трябва да създадат и прилагат политики, които определят кой да има достъп до какви ресурси и при какви обстоятелства. Единна, централизирана система за оторизация е също така и механизъм за лесен одит на достъпа до ресурсите.

еОтор ще реализира контрол на достъпа, управляван от политики, т.нар. Policy Based Access Control (PBAC). PBAC е модел, който има за цел да разреши или откаже достъп до ресурс въз основа на абстрактна политика и изисквания за управление. PBAC съчетава: ресурси; характеристики на заявител заедно с информация за определен набор от обстоятелства, при които е направено искането за достъп; набор от правила, които определят дали този достъп е разрешен или не.



Фигура 19 Логическа схема на PBAC

Система, реализираща PBAC, ще се състои от следните компоненти:

- PEP – Policy Enforcement Point

Това е мястото, където се задействат политиките за достъп. Може да бъде всяка информационна система, в която има защитени ресурси. PEP генерира запитване към PDP за оторизиране на достъпа, като предоставя информация за ресурса и за субекта, който иска достъп до този ресурс.

- PDP – Policy Decision Point

Това е мястото, където се изпълняват всички политики.

- PAP – Policy Authoring Point

Тук се създават и управляват политиките за достъп.

- PIP – Policy Information Point

Тук се съхраняват политиките. Компонентът е отговорен да извлече всички политики, които са свързани със даден субект и ресурс.

По задание всички ресурси в домейна на електронното управление, за които ще се дефинират правила за достъп, ще бъдат регистрирани в Регистър на ресурсите в ДА (виж 7.4). Ресурсите ще се идентифицират еднозначно с помощта на обектен

идентификатор (oid). Обектните идентификатори ще се поддържат в специална схема на обектните идентификатори в БеУ.
Правилата за достъп до ресурси ще описват връзката между: вид ресурс; субект (кой има и съответно кой няма право на достъп); какви действия могат да се извършват от субекта с ресурса; при какви условия и какви действия да се предприемат в случай на отказ или на разрешение за достъп.

Заявката за оторизация ще съдържа:

- Данни за ресурса, до който се осъществява достъп: oid на ресурс (например уеб услуга);
- Данни от издадения от еАвт SAML 2.0 токен със следните данни за заявител:
    - ✓ за физическо лице-заявител на ЕАУ това са: три имена и секторен псевдоним;
    - ✓ за служител в администрация – три имена, секторен псевдоним, oid на администрация, заемана длъжност, адрес на електронна поща и др.;
    - ✓ за ИС на администрация – oid на информационната система, oid на администрацията.

Допълнителни атрибути, необходими за изпълнение на оторизацията, могат да бъдат извличани при необходимост с помощта на компонента Справочник за атрибути (виж 7.7).
еОтор ще поддържа правила за достъп до ресурси за следните групи субекти в електронното управление:

- Администрации и информационни системи - за достъп до ресурси на други администрации;
- Служители в администрациите - за достъп до ресурси в администрациите.

еОтор ще дефинира интерфейс и ще предостави уеб услуга на външни системи за оторизиране на заявки за достъп до системни ресурси.
еОтор ще предостави функционалност за:

- търсене на ресурси по: части от наименованието, по част от oid, по част от наименование на администрация и информационна система и др.;
- дефиниране на политики за определен ресурс или група от ресурси, както и да дефинира набор от възможни действия с тях;
- търсене на политики по идентификатор на ресурс и вид действие.

еОтор ще да бъде реализирана с отворени стандарти.
Предоставяната услуга за оторизиране на достъп до ресурси трябва да бъде реализирана като уеб услуга (SOAP/HTTPS).

еОтор трябва да предоставя уеб базиран потребителски интерфейс за дефиниране на политики за достъп до ресурси.

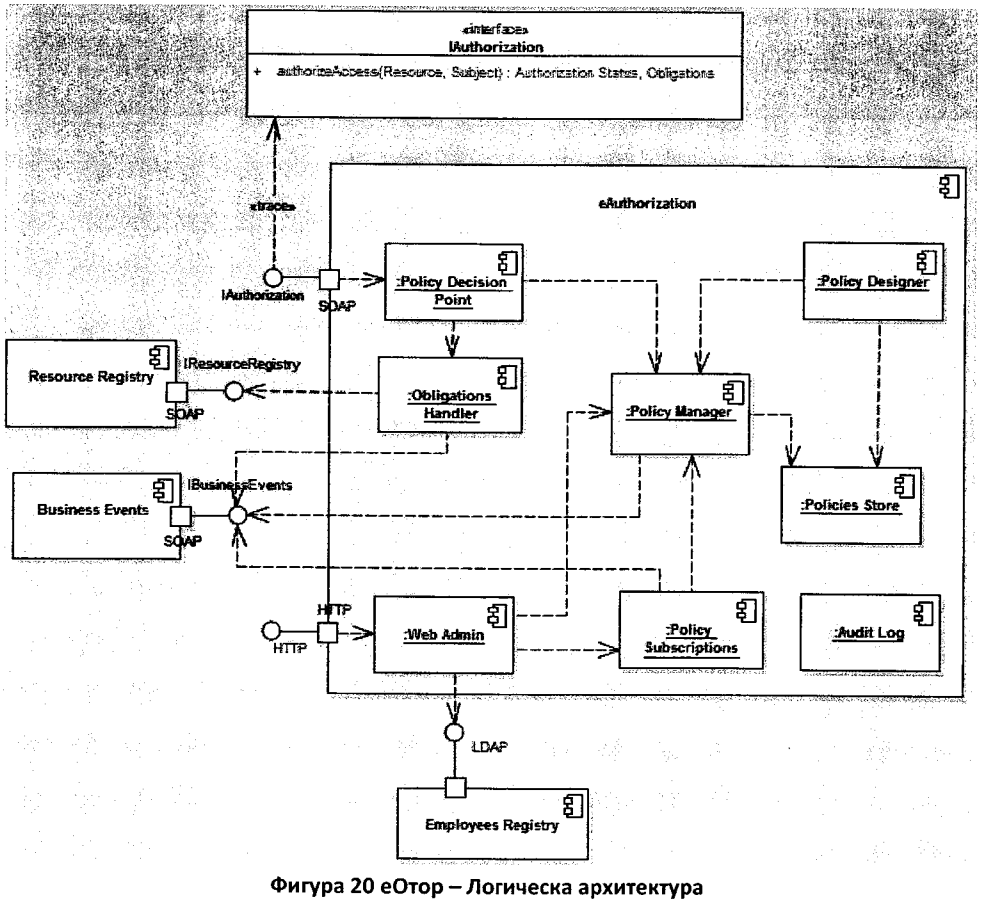
еОтор регистрира всички потребителски действия, свързани с влизане в системата, въвеждане, коригиране и изтриване на данни. Регистрите за одит трябва да съдържат като минимум следните данни: дата и час на влизане в системата и излизане от системата, време на работа, данни за потребителя (само секторен псевдоним, без ЕГН/ЛНЧ и други данни по ЗЗЛД), IP адрес на машината, вид на действията и препратки към извършените промени.

Системата ще предостави механизъм за надеждна защита от хакерски атаки и SQL инжекции.

8.2 Реализация

8.2.1 Компоненти

Логическата структура на еОтор е изобразена на следващата фигура:

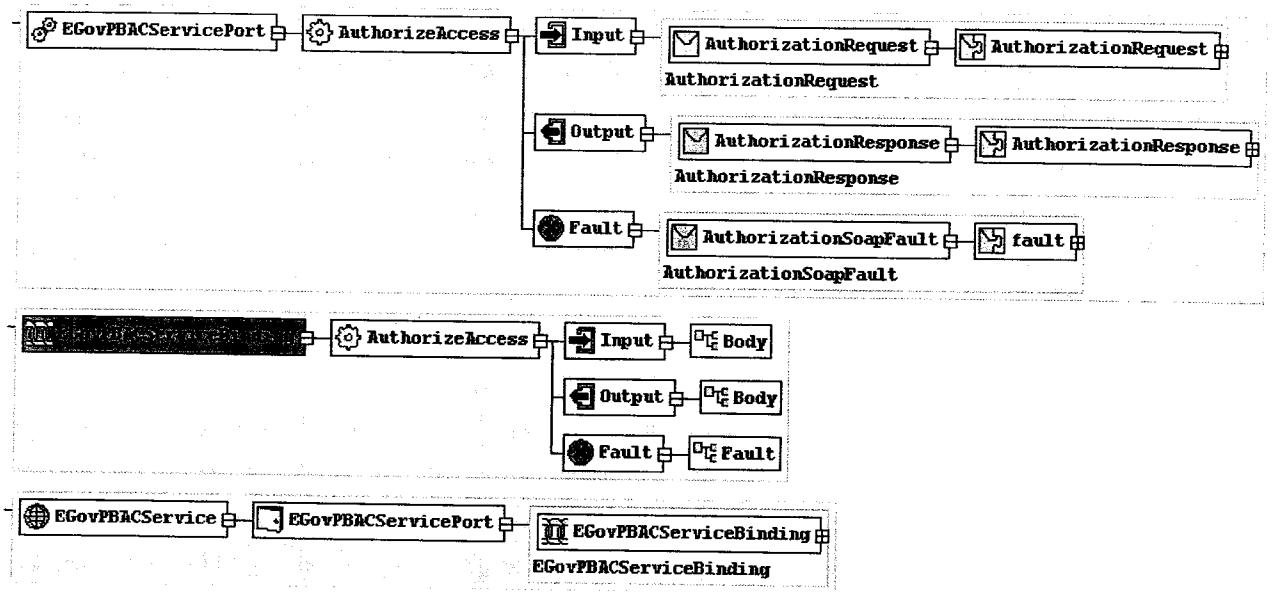


Фигура 20 еОтор – Логическа архитектура

8.2.1.1 Уеб услуга за оторизиране на заявки за достъп до системни ресурси

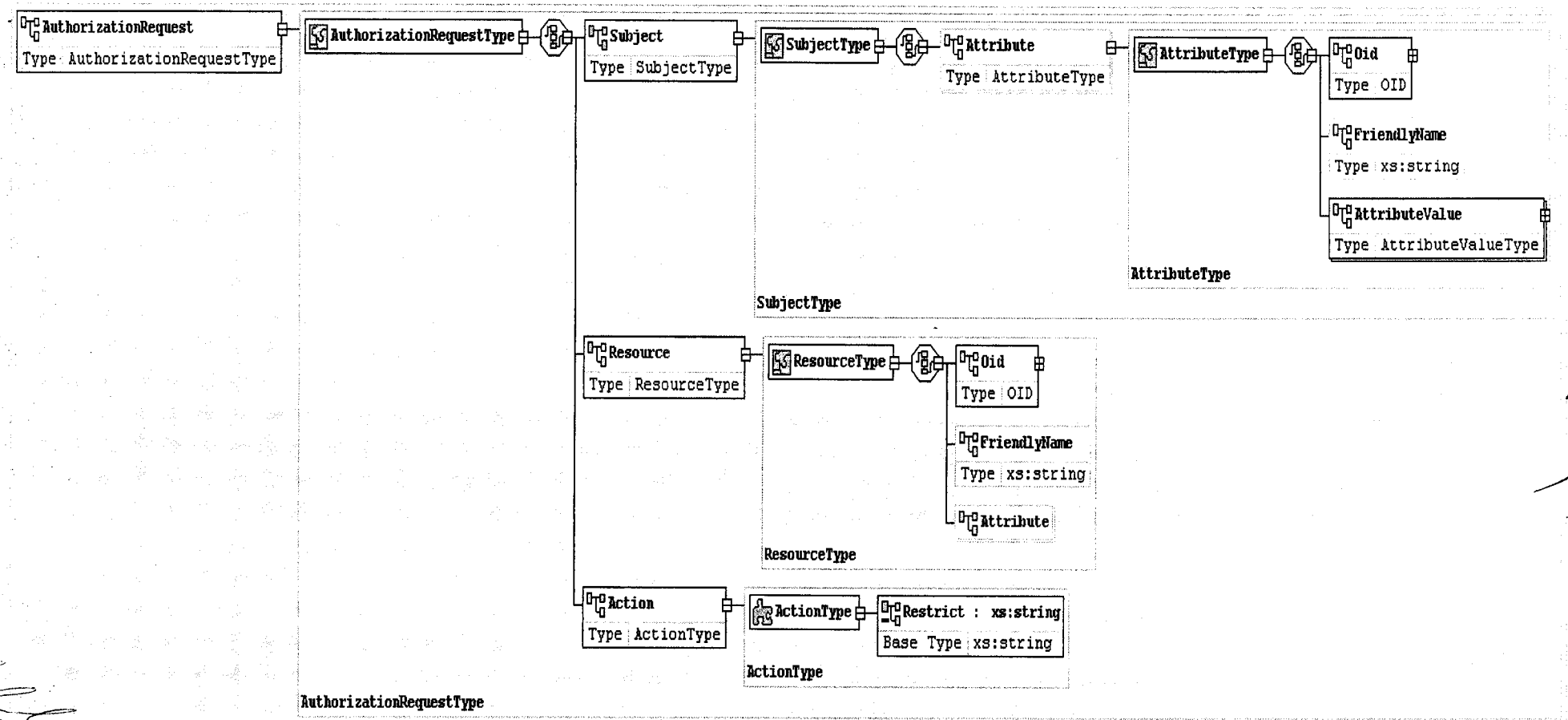
еОтор ще дефинира интерфейс и ще предостави SOAP уеб услуга на външни системи за оторизиране на заявки за достъп до системни ресурси.

Примерна структура на WSDL на уеб услугата е дадена долу:



Услугата ще се казва EGovPBACService и ще бъде достъпна през порта EGovPBACervicePort.

Заявката за оторизация ще използва следната примерна структура на данни:


AuthorizationRequest
Type AuthorizationRequestType
AuthorizationRequestType
Subject
Type SubjectType
SubjectType
Attribute
Type AttributeType
AttributeType
Oid
Type OID
FriendlyName
Type xs:string
AttributeValue
Type AttributeValueType
AttributeType
SubjectType
Resource
Type ResourceType
ResourceType
Oid
Type OID
FriendlyName
Type xs:string
Attribute
ResourceType
Action
Type ActionType
ActionType
Restrict : xs:string
Base Type xs:string
ActionType
AuthorizationRequestType

Примерна заявка, която отговаря на горната схема, е дадена по-долу:

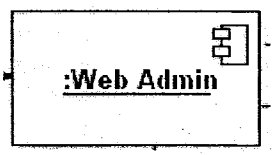
```
<?xml version="1.0"?>
<p1:AuthorizationRequest
xmlns:p1="eauthorization.egov"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">

   <Subject>
      <Attribute>
         <Oid>2.40.2</Oid>
         <FriendlyName>givenName</FriendlyName>
         <AttributeValue DataType="string">
         Ivan
         </AttributeValue>
      </Attribute>

      <Attribute>
         <Oid>2.40.1</Oid>
         <FriendlyName>sureName</FriendlyName>
         <AttributeValue DataType="string">
         Ivanov
         </AttributeValue>
      </Attribute>

      <Attribute>
         <Oid>2.40.2.7</Oid>
         <FriendlyName>Position</FriendlyName>
         <AttributeValue DataType="string">
         Administrator
         </AttributeValue>
      </Attribute>
   </Subject>

   <Resource>
      <Oid>1.2.100.1.......</Oid>
      <FriendlyName>Web Service</FriendlyName>
   </Resource>

   <Action>read</Action>
</p1:AuthorizationRequest>
```

Елементът Subject ще съдържа описание на субекта (физическо лице или информационна услуга), който заявява услугата. В общия случай това са данни , които присъстват в издаден от еАвт SAML токен, например: три имена, ЕГН/ЛНЧ, дата на раждане, администрация, заемана длъжност или oid на информационна система др. Тези данни са достъпни през описанието на елемента `Subject/Attribute`.

Елементът Resource ще съдържа описание на ресурса, достъпът до който трябва да бъде оторизиран. Всеки ресурс в БеУ притежава уникален обектен идентификатор oid.

Елементът Action определя вида на действието, което ще се извършва с ресурса, например: четене (достъп), писане и др.

8.2.1.2 Уеб базирано приложение за поддържане на политиките за достъп



Ще бъде достъпно само за оторизирани служители в ДА, които ще имат право да създават нови и да редактират съществуващи записи в регистъра, както и да променят статуса на ресурс (според процеса за управление на жизнения цикъл).

Потребителският интерфейс на системата ще работи коректно с минимална разделителна способност 1024x768 базирано и ще поддържа следните видове уеб браузъри и версии:

- Microsoft Internet Explorer 8 и по-висока;
- Firefox 24 и по-висока;
- Chrome 22 и по-висока.

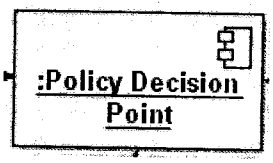
Идентификацията на потребителите ще става с потребителско име и парола. Правата за достъп ще се определят от роли.

Автентикация на потребителите ще се основава на Регистър на служителите в Държавната (виж 7.5).

При промяна на запис приложението ще генерира събитие в Системата за обработка на бизнес събития (виж т.10), така че други системи да бъдат известявани при промени в политиките.

Продуктът IBM ODM предоставя административна конзола за управление на политиките.

8.2.1.3 Policy Decision Point



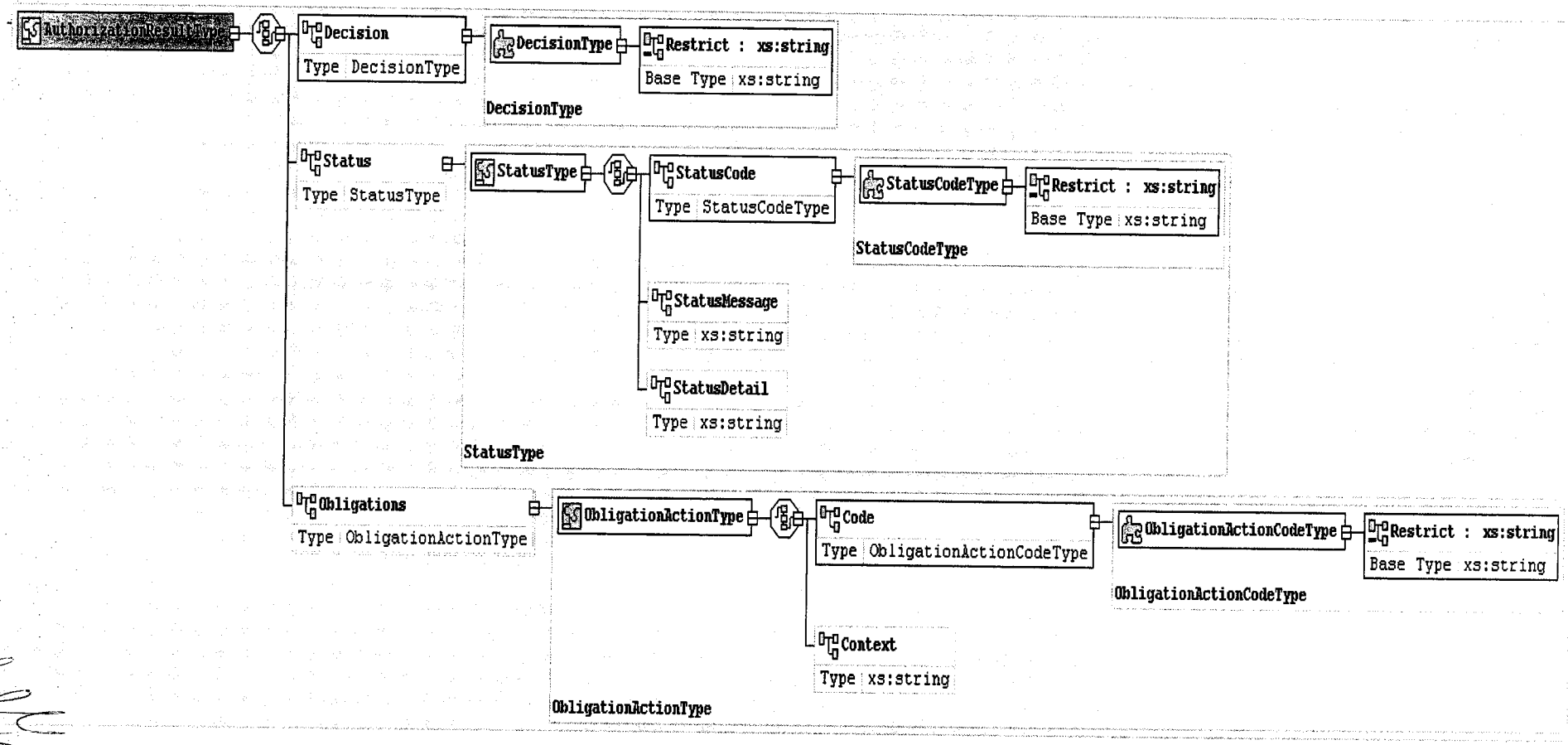
Това е компонентът, отговорен за изпълнение на политиките за достъп. Политиките за достъп ще се реализират като бизнес правила в IBM ODM, описват се със специален DSL (Domain Specific Language) и ще имат следния примерен вид:

```
set 'the resource' to a resource type in the resources of 'the
authorization request' ;
      set 'the subject position in administration' to an attribute type in
the attributes of the subject of 'the authorization request';

if
      the oid of 'the resource' is "1.2.100.1...."
      and the value of 'the subject position in administration' is
"Administrator"
then
      set the decision of 'the authorization result' to "Permit";
      set the status code of the status of 'the authorization result' to
"ok";
      add obligation "send-email", "Authorization was permited" to 'the
authorization result';
else
      set the decision of 'the authorization result' to "Deny";
      set the status code of the status of 'the authorization result' to
"ok";
      add obligation "send-email", "Authorization was denied" to 'the
authorization result' ;
```

В този случай правилото означава, че:

На заявители на услугата, които са служители в администрация със заемана длъжност „Администратор", се разрешава достъпът до ресурс с обектен идентификатор oid: `1.2.100.1....`

Резултатът от заявката за оторизация ще използва следната примерна структура на данни:



Примерен резултат, който отговаря на горната схема, е даден по-долу:

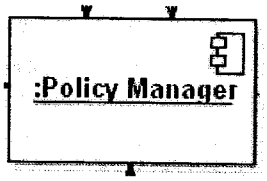
```
<p1:AuthorizationResult xmlns:p1="eauthorization.egov"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
   <Decision>Permit</Decision>

   <Status>
      <StatusCode>ok</StatusCode>
   </Status>

   <Obligations>
      <Obligation>
         <ObligationActions>
            <ObligationAction>
            <Code>emit-event</Code>
                  <Context>… </Context>
            </ObligationAction>
         </ObligationActions>
      </Obligation>
   </Obligations>
</p1:AuthorizationResult >
```

Елементът Decision има две възможни стойности: Permit и Deny – съответно при оторизиран и неразрешен достъп до заявения ресурс.

Елементите Obligations могат да зададат допълнителни действия, които трябва да се предприемат след получаване на отговора, например: известяване на адреса на електронна поща за отказан или разрешен достъп; генериране на бизнес събитие за целите на одита и др.

Компонентът ще се реализира със средствата на IBM ODM Decision Server.

8.2.1.4 Policy Manager



Реализира Governance процеси за бизнес събитията и бизнес правилата.

При промяна на политика за достъп компонентът ще генерира събитие в Системата за обработка на бизнес събития.

Ще се реализира с IBM ODM Decision Center.

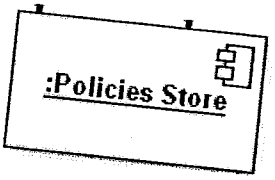
8.2.1.5 Policy Designer

С помощта на редактора за политики за достъп се дефинира мета описание на правилата, създават се нови и променят съществуващи правила.

IBM ODM предоставя две възможности за редакция на политики:

- Редактор на бизнес правила, базиран на платформата Eclipse;
- IBM ODM Decision Center – компонент с уеб базирано приложение за управление на политики;

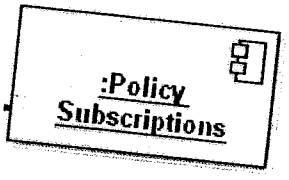
8.2.1.6 Policies Store



Мястото, където се съхраняват политиките за достъп.

Предоставя се от продукта IBM ODM и е релационна СУБД.

8.2.1.7 Policy Subscriptions

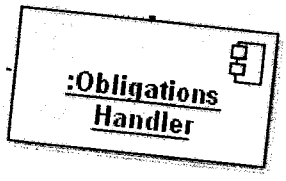


eOтор ще предостави възможност за известяване на заинтересовани администрации, които предоставят електронни услуги, за настъпили промени в регистъра с политики.

Ще бъде реализирана с опашка за съобщения JMS topic, като ще се предостави уеб базиран потребителски интерфейс за абониране за определени видове събития.

Известяването ще бъде конфигурируемо: по адрес на електронна поща или на порт на уеб услуга. За втория случай фирма Бул Ес Ай ще дефинира и предостави подходящ интерфейс.

8.2.1.8 Obligations Handler



Задачата на този компонент е да обработи наличните obligation елементи в резултата от изпълнение на заявката.

Тази функционалност ще се използва в 2 основни случая:

1) При неоторизиран достъп до ресурс („Deny”)

Ще се генерира бизнес събитие.

На основата на това събитие ще се известява на адреса на електронна поща на системен администратор.

Адресът на електронна поща ще се настройва посредством правило за обработка на този вид бизнес събития в продукта IBM ODM.

Obligation елементите ще съдържат информация за създаване и генериране на събитието. Примерна структура на елемента obligations е дадена по-долу и е заимствана от стандарта XACML (eXtensible Access Control Modeling Language):

```
<Obligations>

<Obligation
ObligationId="urn:oasis:names:tc:xacml:example:obligation:email"
FulfillOn="Deny">

<AttributeAssignment
AttributeId="urn:oasis:names:tc:xacml:2.0:example:attribute:text"
```

```
DataType="http://www.w3.org/2001/XMLSch
Достъпът до ресурс беше отказан.
</AttributeAssignment>

<AttributeAssignment
AttributeId="urn:oasis:names:tc:xacml:2
DataType="http://www.w3.org/2001/XMLSch
<SubjectAttributeDesignator
AttributeId="urn:oasis:names:tc:xacml:1
DataType="http://www.w3.org/2001/XMLSche
</AttributeAssignment>
</Obligation>

</Obligations>
```

2) При оторизиран достъп до ресурс („Pern

Obligation елементите ще съдържат информ
събитието. Примерна структура на елемента ob
от стандарта XACML (eXtensible Access Control

```
<Obligations>

<Obligation ObligationId=" urn:egov.bg:pba
FulfillOn="Permit">


<AttributeAssignment
AttributeId="urn:egov.bg:pbac:names:attrib
DataType="urn:egov.bg:concepts:oid">
1.2.100.1.4.5.2.4
</AttributeAssignment>

<AttributeAssignment
AttributeId="urn:oasis:names:tc:xacml:2.0:e
DataType="http://www.w3.org/2001/XMLSchema#
Достъпът до данни в:
</AttributeAssignment>

<AttributeAssignment
AttributeId="urn:oasis:names:tc:xacml:2.0:e
DataType="http://www.w3.org/2001/XMLSchema#.
<SubjectAttributeDesignator
AttributeId="urn:oasis:names:tc:xacml:1.0:su
DataType="http://www.w3.org/2001/XMLSchema#s
/AttributeAssignment>
/Obligation>

/Obligations>
```

bligations Handler ще:

- извлече от Регистъра на ресурсите в ДА
администрацията, информационната система

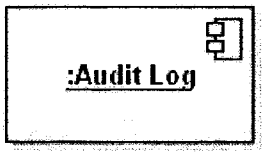
- извлече от subject елемента данни за служителя: три имена, администрация и длъжност.

На основата на тези данни ще се генерира бизнес събитие.

На основата на това събитие ще се създаде запис в Журнала на достъпа до ресурси в БеУ.

Адресът на журнала ще се настройва посредством правило за обработка на този вид бизнес събития в продукта IBM ODM.

8.2.1.9 Audit Log



Този компонент ще регистрира всички потребителски действия, свързани с влизане в системата, въвеждане, коригиране и изтриване на данни. Ще се записват най-малко следните данни: дата и час на влизане в системата и излизане от системата, време на работа, данни за потребителя (само секторен псевдоним, без ЕГН/ЛНЧ и други данни по ЗЗЛД), IP адрес на машината, вид на действията и препратки към извършените промени.

Ще бъде реализиран със средствата за логване на продукта IBM ODM.

9 МЕТОДИКА ЗА РЕАЛИЗАЦИЯ НА РАЗВИТИЕТО НА ШИНАТА ЗА УСЛУГИ (ESB) ЗА ВРЪЗКА С КОМПОНЕНТИТЕ ЗА ЕДНОКРАТНА АВТЕНТИКАЦИЯ И ЕЛЕКТРОННА ОТОРИЗАЦИЯ

9.1 Описание

Шината за услуги (ESB) е интеграционният компонент в БеУ, през който ще преминават всички заявки за изпълнение на уеб услуги в БеУ.

Понастоящем ESB в БеУ е реализирана с IBM WebSphere ESB (IBM WESB). Програмният модел на IBM WESB се основава на спецификацията Service Component Architecture (SCA) на организацията OASIS.

За нуждите на БеУ ще се разработят 2 компонента от вид mediation primitive – по един за интеграцията с еАвт и еОтор.

В случая на интеграция с еАвт задачата на този компонент е да се обръща към уеб услуга (SOAP/HTTPS), предоставяна от компонента еАвт за валидиране на SAML токени, които идентифицират заявител на електронна услуга. Компонентът ще генерира заявка според спецификацията WS-Trust Validation binding.

Получавайки резултат от запитването, mediation primitive компонентът:

- при невалиден SAML токен – спира изпълнението на заявката, генерира събитие в Системата за обработка на бизнес събития и генерира SOAP Fault с описание на грешката;
- при валиден SAML токен – продължава с изпълнението на заявката.

Mediation primitive компонентът ще извлича адреса на еАвт от регистъра с уеб услуги UDDI. Това ще гарантира, че евентуална промяна на крайния адрес на еАвт няма да се отрази на функционирането на шината за услуги (ESB).

Mediation primitive компонентът ще реализира изходен терминал, който се активира при валиден токен.

Mediation primitive компонентът ще реализира изходен терминал, който се активира при невалиден токен.

Mediation primitive компонентът ще реализира изходен терминал, който се активира при грешка в изпълнението, например SOAP Fault на услугата, предоставяна от еАвт или друга програмна грешка.

В случая на интеграция с еОтор задачата на този компонент е да се обръща към уеб услуга (SOAP/HTTPS), предоставяна от компонента еОтор за оторизиране на достъпа до уеб услуга. Заявката трябва да съдържа обектен идентификатор на ресурс, за който

се изисква достъп; субект, който иска достъп; както и действие, за което се изисква оторизиране, например: четене, запис, премахване и др.

Тази информация ще се съдържа в SAML токен в SOAP заявката, изпратена към IBM WESB.

Получавайки резултат от запитването, mediation primitive компонентът:

- При оторизиран достъп - пренасочва заявката към съответния ресурс (в общия случай това е порт на уеб услуга);
- При отказан достъп - генерира събитие в системата за генериране и обработка на бизнес събития.

Mediation primitive компонентът ще извлича адреса на еОтор от регистъра с уеб услуги UDDI. Това ще гарантира, че евентуална промяна на крайния адрес на еОтор няма да се отрази на функционирането на шината за услуги (ESB).

Mediation primitive компонентът ще реализира изходен терминал, който се активира при оторизиран достъп.

Mediation primitive компонентът ще реализира изходен терминал, който се активира при НЕоторизиран достъп.

Mediation primitive компонентът ще реализира изходен терминал, който се активира при грешка в изпълнението, например SOAP Fault на услугата, предоставяна от еОтор или друга програмна грешка.

9.2 Реализация

9.2.1 Използвани технологии и технически средства

ESB осигурява набор от инфраструктурни възможности, който позволява интегрирането на услуги в ориентирана към услугите архитектура (SOA). ESB предоставя единна точка на управление на разпределена среда за комуникация между заявителите и доставчици на услуги. Тези доставчици могат да включват функционални услуги, предоставяни от съществуващите приложения.

В случая наличната при Възложителя платформа за ESB базирана на IBM WebSphere ESB покрива поставените изисквания. IBM WebSphere ESB освен, че е налична при възложителя, се основава на общоприети индустриални стандарти, позволява унифициран и гъвкав подход при реализирането на интеграция между разнородни системи, като опростява цялостния процес на интеграция и намалява зависимостта между отделните компоненти.

9.2.1.1 IBM WESB

Основни характеристики на IBM WebSphere ESB са и още:

- предоставя индустриални интерфейси и протоколи като по този начин намалява цената и риска при въвеждането на промени в архитектурата и изграждането на нови системи и решения;
- позволява инкременталното свързване на системи и услуги, така че да се намалят разходите за интеграция и да се предостави възможност за повторно използване на модулите и услугите като се раздели бизнес логиката на приложенията от задачите по интеграция;
- осигурява централно управляема среда, в която услугите да могат да бъдат променяни или добавяни без да е необходимо да се спира системата;
- осигурява среда за интеграция, която да премахне зависимостта от платформите и архитектурите на отделните системи, които ще се интегрират сега или в бъдеще, като намали същевременно броя и комплексността на интерфейсите между тях;
- осигурява обмен на транзакции, данни и съобщения между системите дори и когато някоя система или мрежа е недостъпна.

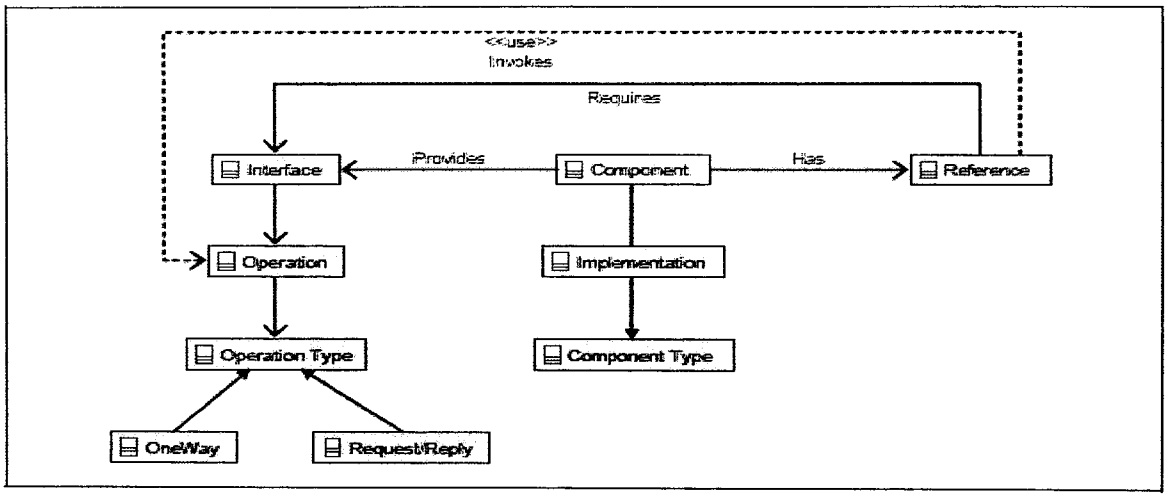
9.2.1.1.1 Основни понятия

IBM WebSphere ESB се основава на спецификацията Service Component Architecture (SCA) на организацията OASIS и реализира SCA ядро, което предоставя java програмен интерфейс и инфраструктурни компоненти, необходими за поддръжка на SCA компоненти.

Концепцията за SCA се основава около използването на компоненти, които предоставят услуги. Тези услуги могат да се извикват от външни системи или други компоненти. Услугите се описват посредством интерфейси, които дефинират данните, необходими за извикване на методите от интерфейса. Всеки интерфейс е дефиниран като WSDL порт на уеб услуга или като java интерфейс.

SCA компоненти могат да използват услуги от други SCA компоненти. От гледна точка на потребител на услуги (service requester) един SCA компонент представлява черна кутия. Начинът на реализация на услугата остава скрит за извикващата страна.

Един интерфейс може да има няколко операции. Операциите могат да бъдат еднопосочни (one-way) или двупосочни (two-way). На следващата UML диаграма е изобразен опростен модел на архитектурата на SCA:

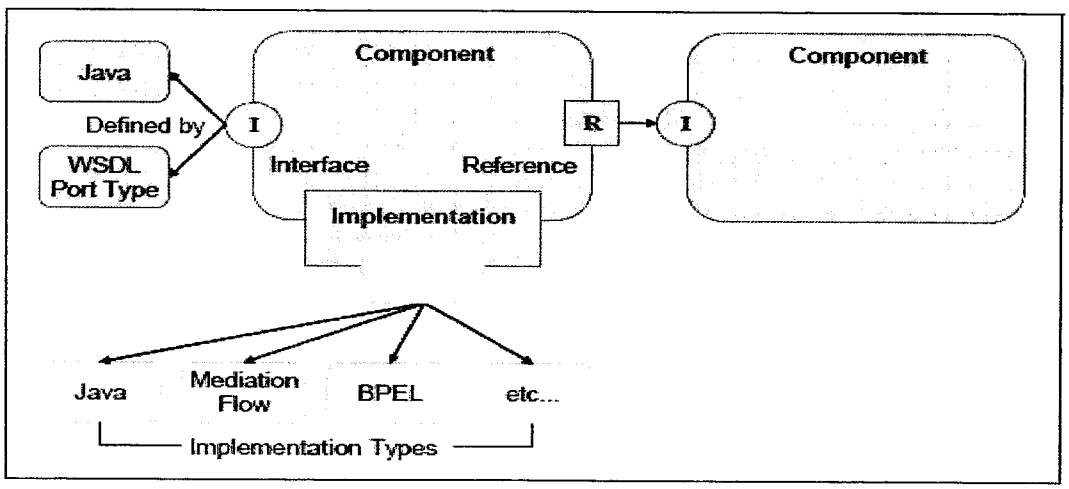


SCA компонентите могат да имат няколко варианта за реализация, като: Java, BPEL, C++.

SCA компонентите в WebSphere ESB могат да се „снаждат" един с друг в т. нар. mediation модул. IBM WebSphere ESB предоставя специализиран SCA компонент, наречен mediation flow.

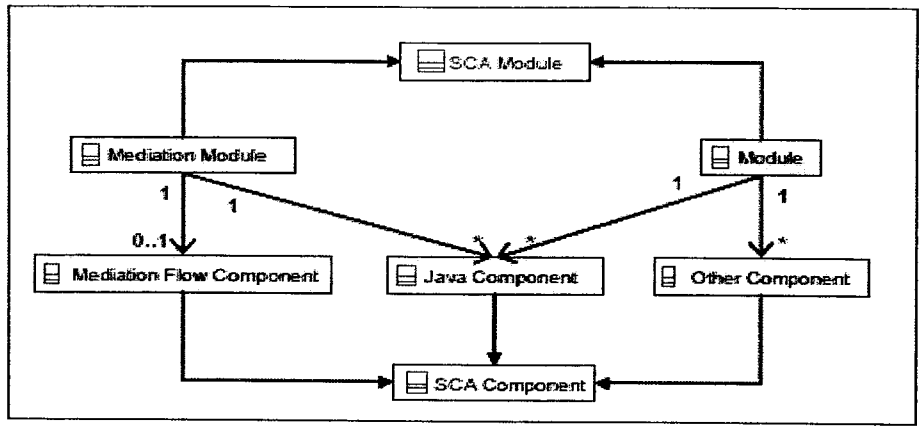
Един mediation модул се пакетира като Java Enterprise Archive (EAR) файл, който се инсталира в средата на IBM Websphere ESB.

Логическата архитектура на SCA компонент е дадена на следващата фигура:



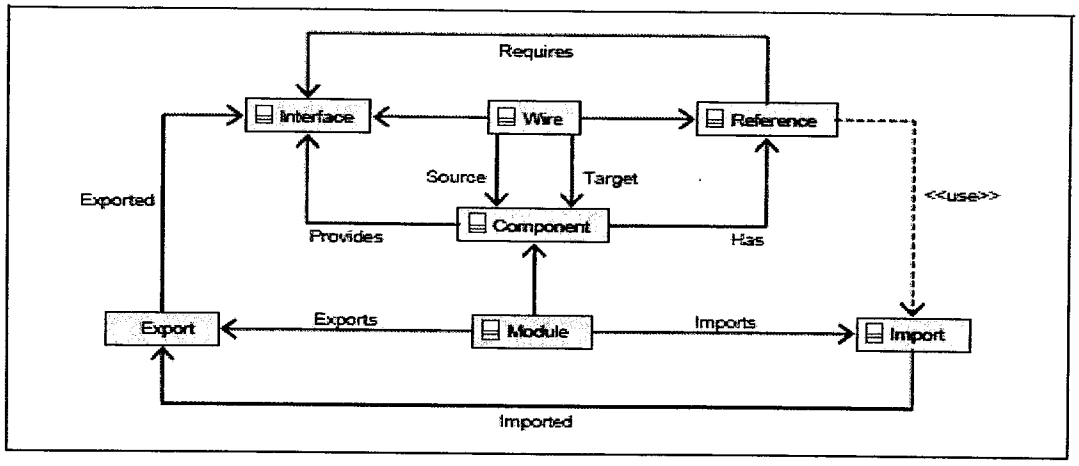
9.2.1.1.1.1 SCA модули

SCA модулите в IBM WebSphere ESB се наричат *mediation* модули. На следващата UML диаграма е показана връзката между mediation и основните SCA концепции:



Модулите предоставят т. нар. експорти (exports), които могат да бъдат извиквани от други SCA модули посредством т. нар. импорти (imports) или от клиент – външен за SCA модела.

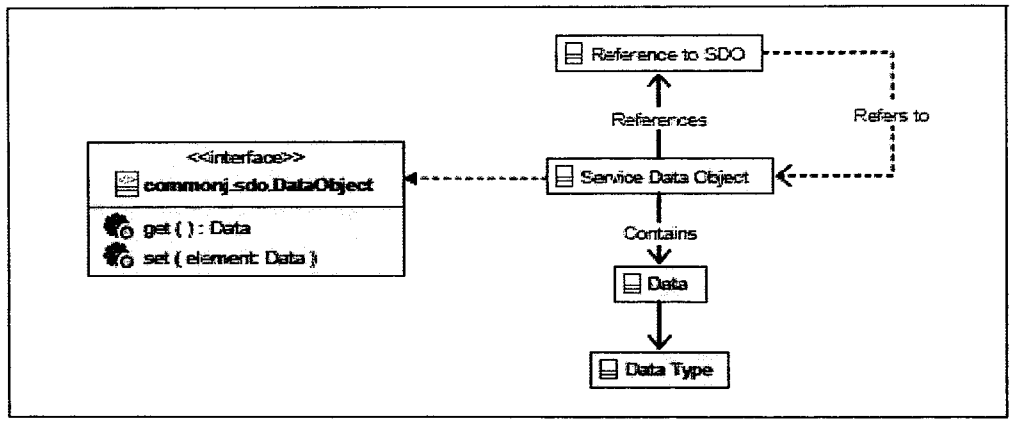
На следващата фигура е изобразен опростен UML модел на SCA модул с връзките между референции, интерфейси, експорти и импорти:



9.2.1.1.1.2 Service Data Objects (SDO)

Service Data Objects (SDO) са основният начин за обмен на данни в SCA. Интерфейсът на всеки SDO е разширява интерфейса дефиниран в commonj.sdo.DataObject.

SDO служат за пренос на данни и референции към други обекти, което е изобразено на следващата фигура:

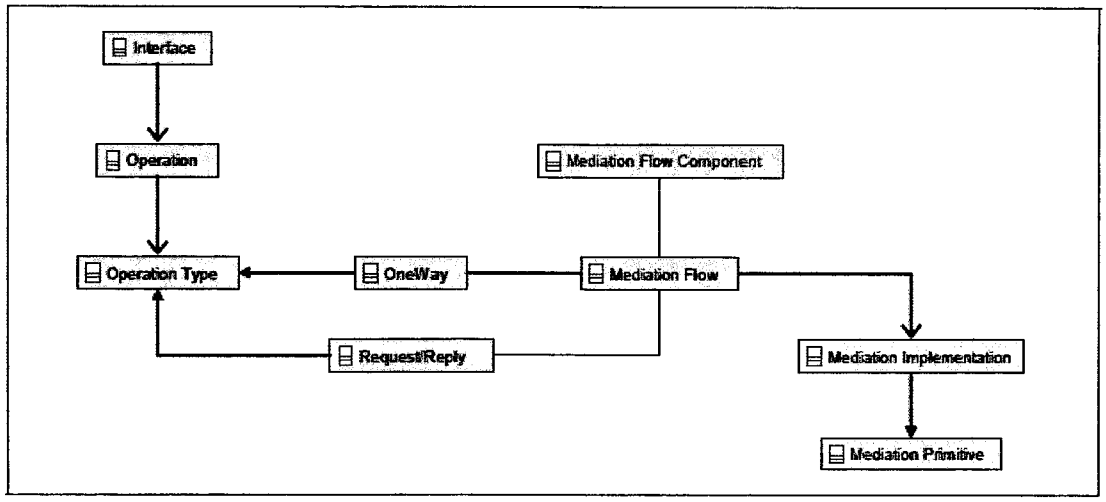


9.2.1.1.1.3 *Mediation модул*

Един mediation модул представлява вид SCA компонент и съдържа поне един компонент mediation flow (виж по-долу).

9.2.1.1.1.4 *Mediation flow*

Mediation flow компонентите съдържат единичен процес на обработка (*mediation flow)* за всяка операция от интерфейса на компонента. За еднопосочните операции (one-way) се дефинира request flow, а за двупосочните операции (two-way) имаме request flow и response (reply) flow, както е изобразено на следващата фигура:

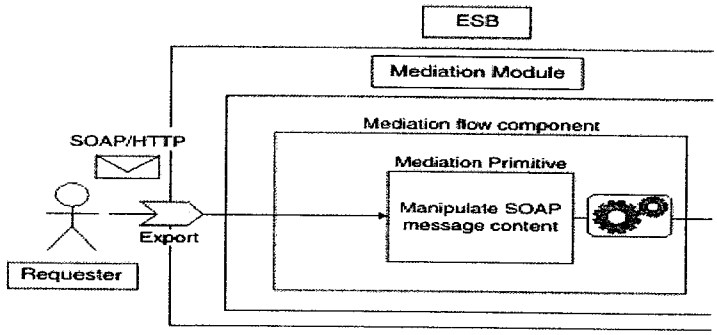


Както се вижда от горната фигура, mediation flow се състои от няколко mediation примитиви, които са свързани помежду си. Всеки примитив реализира специфична функционалност. Съществуват следните примитиви:

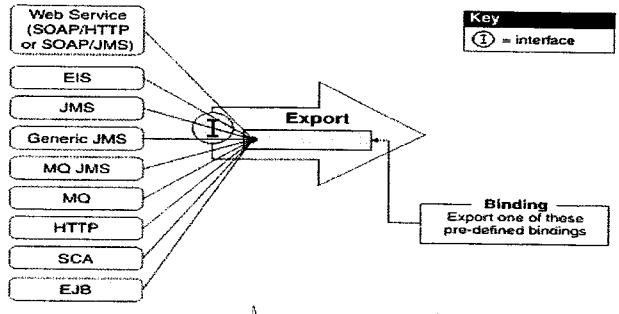
- XSLT – Този примитив служи за трансформиране на данни от един формат в друг.
- Message Element Setter – Този примитив позволява елементи от съобщението да бъдат създавани, изтривани или копирани от други елементи в същото съобщение.
- Message Filter – Този примитив позволява филтриране на съобщения по съдържание и се основава на XPath. Съобщението може да бъде пренасочвано към различни крайни точки (terminals) в зависимост от съдържанието.

- Message Logger – Този примитив позволява съобщението да бъде записано в база данни с цел последващ одит.
- Database Lookup – Този примитив позволява извличане на данни от база данни и включването им в съобщението. Извличането на информация става на основата на ключ, който е част от самото съобщение.
- Event Emitter – Този примитив позволява генерирането на събития през предоставената от IBM WebSphere Application Server среда Common Event Infrastructure (CEI).
- Endpoint Lookup – Този примитив позволява извличане на адреса на услуга от WebSphere Services Registry and Repository.
- Fail – Този примитив принуждава един mediation flow да спре изпълнението и да изпълни roll back на текущата трансакция. В случая, когато mediation компонентът предоставя уеб услуга , резултатът ще е връщане на SOAP fault съобщение.
- Stop - Този примитив принуждава един mediation flow да спре изпълнението.
- Custom - Този примитив позволява реализация на специфична функционалност с използването на java code.

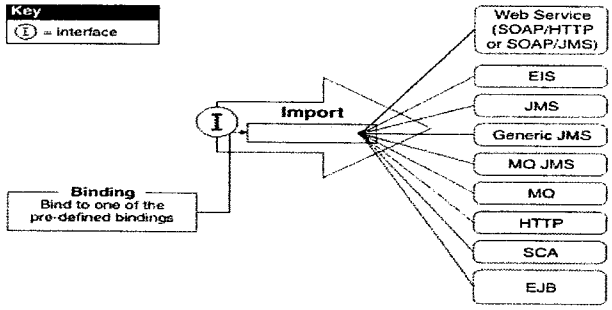
Други SCA модули и/или външни системи могат да обменят информация с mediation модул посредством т. нар. *Експорт* (Export). Един mediation модул може да извиква операции от други SCA модули и/или външни системи посредством т. нар. *Импорт* (Import) (виж следващата фигура):



Един *Експорт* може да има следните начини на свързване (binding):



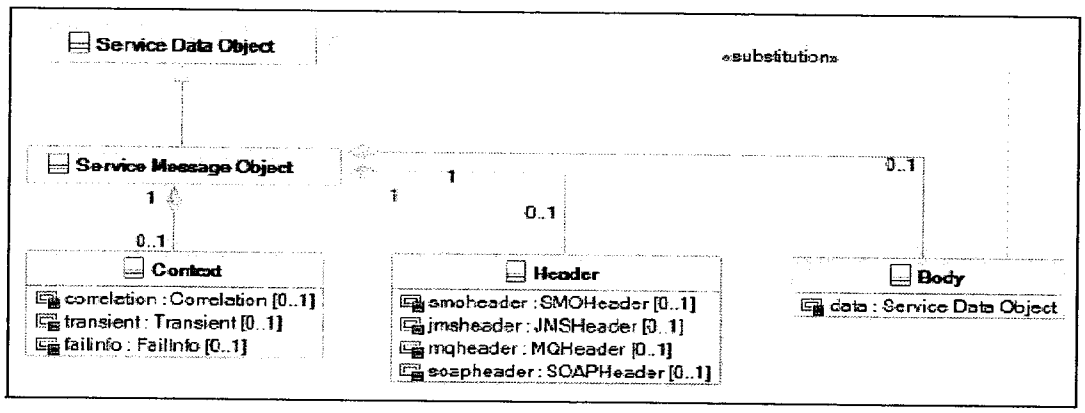
Един *Импорт* може да има следните начини на свързване (binding):



9.2.1.1.1.5 Service Message Objects (SMO)

Обменът на информация между отделните примитиви в един mediation flow се осъществява с помощта на Service Message Objects (SMO). SMO се създават автоматично, когато се създаде инстанция на mediation flow.

SMO представляват вид SDO със следната структура:



Основните елементи на SMO са:

Context - състои се от 3 секции:

- correlation – използва се за съхранение на информация, която служи за съотнасяне на съобщения при двупосочните операции. Когато се получи reply съобщение, съответният response flow автоматически има достъп до същият контекст на корелация;
- transient – използва се за съхранение на информация, която е достъпна само в рамките на еднопосочна операция.
- faiInfo – използва се за съхраняване на информация за грешки, възникнали по време на изпълнение на mediation flow

Header –съдържа информация за операцията (име, вид: еднопосочна, двупосочна и др., информация за протокола на извикване: JMS, MQ, SOAP)

Body – съдържа данните като SDO обект.

9.2.1.1.1.6 SCA свързване (binding)

SCA свързване (binding) на интерфейс на компонент определя протокол, адрес и политики на реализацията на интерфейса.

IBM WebSphere ESB предоставя следните видове свързване:

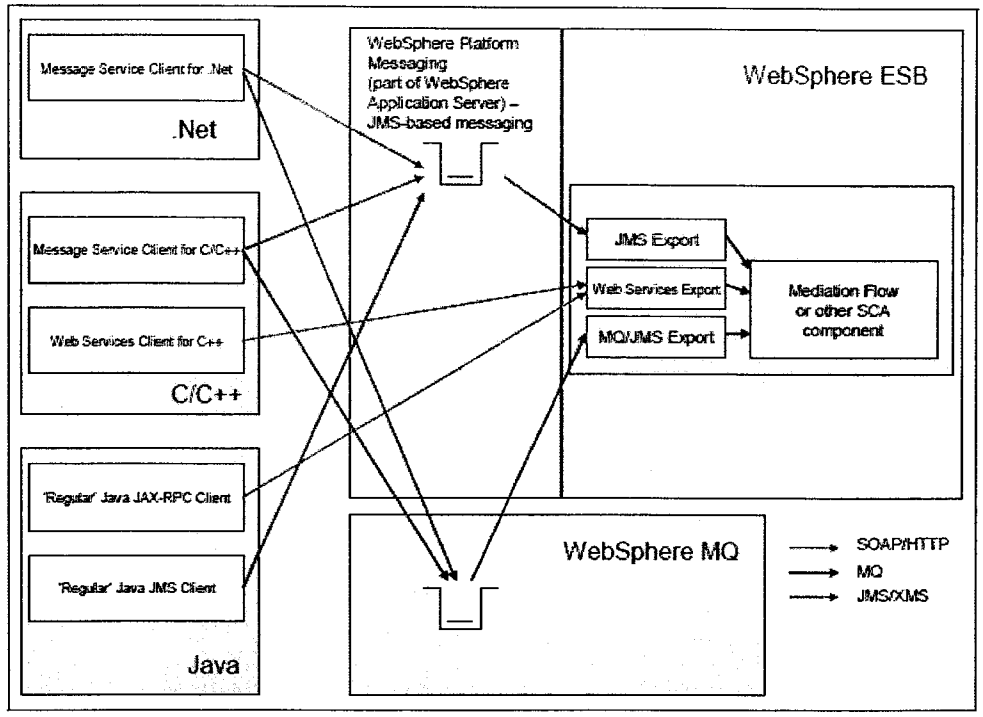
- SOAP/HTTP - за връзка със системи по уеб услуги;
- SOAP/JMS - за връзка със системи по JMS;
- MQ и MQ/JMS - за връзка със системи, реализирани на основата на WebSphere MQ;
- Stateless session EJB - за връзка със системи по RMI протокол;
- EIS - за връзка със системи с помощта на т. нар. адаптери. Предоставят се следните адаптери:
  FTP – използва се за обмен на информация по ftp протокол.
  JDBC - използва се за обмен на информация по ftp протокол през интерфейсни таблици.
  E-mail - използва се за обмен на информация с помощта на електронни съобщения.
  Flat File - използва се за обмен на информация с помощта на файлове с предварително дефинирана структура.

9.2.1.1.1.7 Клиенти

Достъпът до IBM WebSphere ESB освен по стандартните механизми, описани досега в техническото предложение, може да става и през следните предоставяни клиенти:

- Message Service клиент за C/C++ - предоставя JMS подобен интерфейс за приложения, програмирани на C/C++;
- Message Service клиент за.Net - предоставя JMS подобен интерфейс за приложения, програмирани на .Net платформа;
- Web services клиент за C++ - позволява достъп до уеб услуги, предоставяни през ESB, за приложения на C++;
- Java JAX-RPC клиент – предоставя клиент за java приложения за достъп до уеб услуги, предоставяни през ESB
- Java JMS клиент - предоставя JMS интерфейс за достъп до ESB за приложения, програмирани на java.

Видовете клиенти и начинът на комуникация с IBM WebSphere ESB е изобразен на следващата фигура:



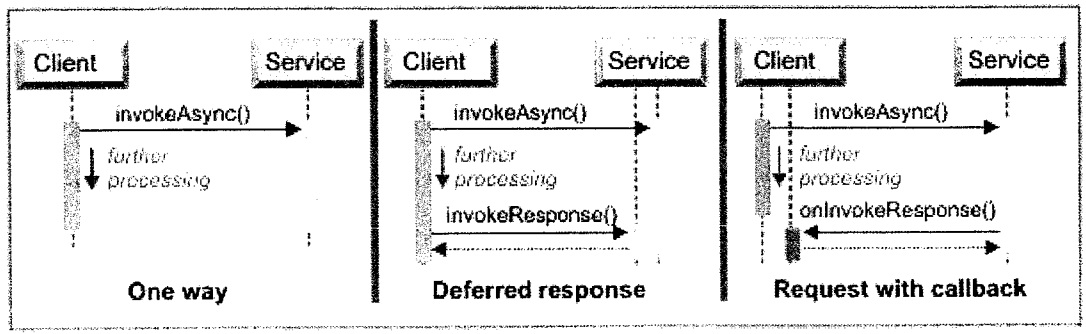
9.2.1.1.1.8 Начини на извикване на SCA компоненти

Операциите, дефинирани в интерфейсите на SCA компоненти, могат да се извикват синхронно и асинхронно, независимо от начина на реализация на компонента (също: синхронна или асинхронна). За правилното функциониране се грижи средата на IBM WebSphere ESB.

Предоставят следните стилове на извикване:

- Синхронно (synchronous);
- Асинхронно с еднопосочна операция (asynchronous using one-way operation);
- Асинхронно с предоставяне на callback обект (asynchronous with callback);
- Асинхронно с е с отложен отговор (asynchronous with deferred response).

Особеностите на асинхронните стилове на извикване са дадени на следващата фигура:



9.2.1.1.1.9 UDDI

UDDI (Universal, Description, Discovery and Integration) е платформено независим регистър на услуги, предлагани от информационни системи. Услугите са категоризирани според поддържани таксономии. UDDI е отворена инициатива, позволяваща на различните видове бизнес да публикуват списъци с услуги, да се намират лесно и да дефинират как дадена услуга или софтуерни приложения взаимодействат с интернет. UDDI е един от основните стандарти за уеб услуги. Разработен е да бъде използван от SOAP съобщения и да предоставя достъп до WSDL (Web Services Description Language) документи, описващи протоколни свързвания и формати за съобщения, изисквани за взаимодействие с уеб услуги.

В инфраструктурата на БеУ има инсталиран регистър UDDI регистър, в който са ще се дефинират всички публично достъпни уеб услуги, предлагани от администрациите.

Комбинацията от ESB и UDDI позволява реализирането на т. нар. виртуализиране на услуги. В регистъра се пази връзката между система, УРИ на АИС и точката за достъп. По този начин при инсталиране на нова АИС и/или преместване на някоя система на нов адрес, се променя само точката за достъп (access point) в UDDI регистъра без да е необходимо пренаписване на части от приложенията

9.2.2 Интеграция с еАвт

В случая на интеграция с еАвт задачата на този компонент е да се обръща към уеб услуга (SOAP/HTTPS), предоставяна от компонента еАвт за валидиране на SAML токени, които идентифицират заявител на електронна услуга. Компонентът ще генерира заявка според спецификацията WS-Trust Validation binding.

Получавайки резултат от запитването, mediation primitive компонентът:

- При невалиден SAML токен – спира изпълнението на заявката, генерира събитие в Системата за обработка на бизнес събития и генерира SOAP Fault с описание на грешката;
- При валиден SAML токен – продължава с изпълнението на заявката.

Mediation primitive компонентът ще извлича адреса на еАвт от регистъра с уеб услуги UDDI. Това ще гарантира, че евентуална промяна на крайния адрес на еАвт няма да се отрази на функционирането на шината за услуги (ESB.

Mediation primitive компонентът ще реализира изходен терминал, който се активира при валиден токен.

Mediation primitive компонентът ще реализира изходен терминал, който се активира при НЕвалиден токен.

Mediation primitive компонентът ще реализира изходен терминал, който се активира при грешка в изпълнението, например SOAP Fault на услугата, предоставяна от еАвт или друга програмна грешка.

9.2.3 Интеграция с еОтор

Компонентът за изпълнение на политиките за достъп до ресурси или Policy Enforcement Point (PEP) според PBAC в ESB ще бъде реализиран като mediation primitive.

Задачата на този компонент е да се обръща към уеб услуга (SOAP/HTTPS), предоставяна от компонента еОтор за оторизиране на достъпа до уеб услуга. Заявката трябва да съдържа обектен идентификатор на ресурс, за който се изисква достъп; субект, който иска достъп; както и действие, за което се изисква оторизиране, например: четене, запис, премахване и др.

Тази информация ще се съдържа в SAML токен в SOAP заявката, изпратена към IBM WESB.

Получавайки резултат от запитването, mediation primitive компонентът:

- При оторизиран достъп - пренасочва заявката към съответния ресурс (в общия случай това е порт на уеб услуга);
- При отказан достъп - генерира събитие в системата за генериране и обработка на бизнес събития.

Mediation primitive компонентът ще извлича адреса на еОтор от регистъра с уеб услуги UDDI. Това ще гарантира, че евентуална промяна на крайния адрес на еОтор няма да се отрази на функционирането на шината за услуги (ESB).

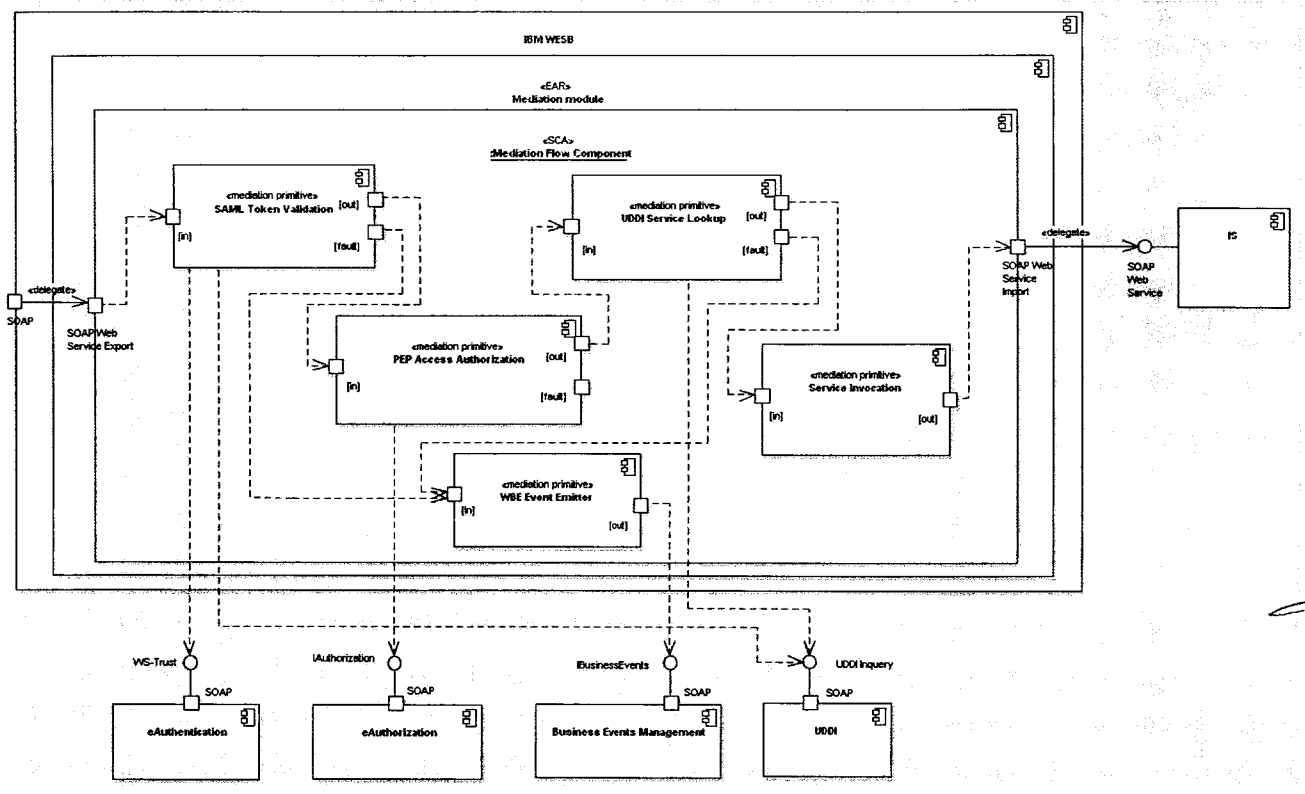
Mediation primitive компонентът ще реализира изходен терминал, който се активира при оторизиран достъп.

Mediation primitive компонентът ще реализира изходен терминал, който се активира при НЕоторизиран достъп.

Mediation primitive компонентът ще реализира изходен терминал, който се активира при грешка в изпълнението, например SOAP Fault на услугата, предоставяна от еОтор или друга програмна грешка.

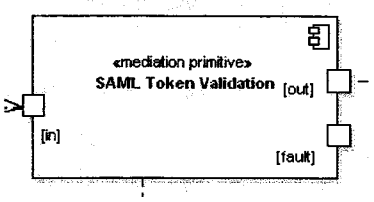
9.2.4 Интеграция на mediation primitive компонентите в ESB

Основните компоненти, участващи е реализацията, са изобразени на следващата фигура:



Фигура 21 Интеграция на mediation primitive компонентите в ESB

9.2.4.1 Компонент за интеграция с еАвт



Компонентът за интеграция с еАвт (КИеАвт) е mediation primitive компонент, който е отговорен за валидиране на SAML токени в средата на IBM WESB.

Компонентът има един входен терминал [in], един изходен терминал [out] и един терминал за сигнализиране на грешка при изпълнение [fault].

SAML токенът, както и всички елементи в елемента soap:Header, са достъпни в SMO обект, който се получава на входния [in] терминал.

Пример за SOAP заявка, която преминава през ESB е дадена по-долу:

```
<SOAP:Envelope xmlns:SOAP="...">
  <SOAP:Header>
    <wsse:Security xmlns:wsse="...">
      <saml:Assertion
        xmlns:saml="..."
        AssertionID="..."
        IssueInstant="..."
        Issuer="..."
        MajorVersion="1"
        MinorVersion="1">
      <saml:AuthenticationStatem
          <saml:Subject>
              <saml:NameID
                  Format="..."
                  Name="urn:oid:2.5.4.45"
                  FriendlyName="uniqueIdentifier">
                    ЕГН/ЛНЧ
```

SAML токен, издаден от еАвт.

Съдържа данни за заявителя – в случая това е физическо лице.

Токенът е достъпен в SMO

```
            </saml:NameID>
          <saml:SubjectConfirmation>
            <saml:ConfirmationMethod>
                urn:oasis:names:tc:SAML:1.0:cm:sender-vouches
            </saml:ConfirmationMethod>
          </saml:SubjectConfirmation>
        </saml:Subject>
      </saml:AuthenticationStatement>
    </saml:Assertion>
  </wsse:Security>
</SOAP:Header>
<SOAP:Body>
....MESSAGE....
</SOAP:Body>
</SOAP:Envelope>
```

КИеАвт извлича SAML токена от SMO, след което КИеАвт извлича актуалния адрес на уеб услугата, предоставяна от еАвт, от Регистъра за уеб услуги (UDDI) по обектен идентификатор (oid) на услугата.

В UDDI за всяка услуга в businessService има дефинирани класификатори (tModel), които позволяват каталогизиране на услугите. Един от тези класификатори е уникалния обектен идентификатор (oid) според схемата на обектините идентификатори в БеУ.

След получаване на адреса на услугата на еАвт, КИеАвт генерира заявка за валидиране на токен към еАвт. За целта се използва WS-Trust Validation Binding. Примерна заявка за валидиране на SAML токен ще има следния вид:

```
<wst:RequestSecurityToken xmlns:wst="...">
    <wst:TokenType>
       http://docs.oasis-open.org/ws-sx/ws-trust/200512/RSTR/Status
    </wst:TokenType>
    <wst:RequestType>
        http://docs.oasis-open.org/ws-sx/ws-trust/200512/Validate
    </wst:RequestType>
    <wst:ValidateTarget>
            SAML токен
    </wst:ValidateTarget>
</wst:RequestSecurityToken>
```

Елементът `/wst:RequestSecurityToken/wst:ValidateTarget` съдържа SAML токена, който трябва да бъде валидиран.

еАвт получава заявката, проверява валидността на SAML токена и връща следния резултат:

```
<wst:RequestSecurityTokenResponse xmlns:wst="..."  >
   <wst:TokenType>...</wst:TokenType>
   <wst:RequestedSecurityToken>...</wst:RequestedSecurityToken>
...
   <wst:Status>
      <wst:Code>...</wst:Code>
```

```
      <wst:Reason>...</wst:Reason>
   </wst:Status>
</wst:RequestSecurityTokenResponse>
```

КИеАвт определя резултата от валидацията:

1) При валиден SAML токен

Елементът `/wst:RequestSecurityTokenResponse/wst:Status/wst:Code` съдържа следната стойност: `http://docs.oasis-open.org/ws-sx/ws-trust/200512/status/valid`

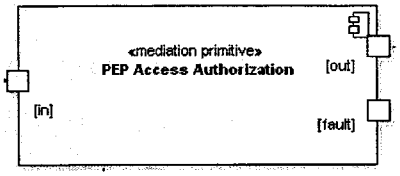
В този случай КИеАвт продължава с изпълнението на заявката.

2) При невалиден SAML токен

Елементът `/wst:RequestSecurityTokenResponse/wst:Status/wst:Code` съдържа следната стойност: `http://docs.oasis-open.org/ws-sx/ws-trust/200512/status/invalid`

В този случай КИеАвт спира изпълнението на заявката, задейства [fault] терминала, което предизвиква генериране на събитие в Системата за обработка на бизнес събития презн генерира SOAP Fault с описание на грешката.

9.2.4.2 Компонент за интеграция с еОтор



Компонентът за интеграция с еОтор (КИеОтор) е mediation primitive компонент, който е отговорен за оторизиране на достъп до уеб услуги, предоставяни от администрации, в средата на IBM WESB.

Компонентът има един входен терминал [in], един изходен терминал [out] и един терминал за сигнализиране на грешка при изпълнение [fault]. Този компонент ще е отговорен за извличане на необходимата информация за субекта от SAML токена в заявката за уеб услуга (SAML токенът, както и всички елементи в елемента soap:Header, са достъпни в SMO обект, който се получава на входния [in] терминал), създаване на заявка към Компонента за електронна оторизация (еОтор) и получаване на отговор.

КИеОтор извлича от UDDI актуалния адрес на уеб услугата на еОтор и генерира завка от вида:

```
<SOAP:Envelope xmlns:SOAP="...">
   <SOAP:Header>
      <wsse:Security xmlns:wsse="...">
      </wsse:Security>
   </SOAP:Header>

<SOAP:Body>
<p1:AuthorizeAcess>
xmlns:p1="eauthorization.egov"
```

```
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
   <Subject>
      <Attribute>
         <Oid>2.40.2</Oid>
         <FriendlyName>givenName</FriendlyName>
         <AttributeValue DataType="string">
         Ivan
         </AttributeValue>
      </Attribute>

      <Attribute>
         <Oid>2.40.1</Oid>
         <FriendlyName>sureName</FriendlyName>
         <AttributeValue DataType="string">
         Ivanov
         </AttributeValue>
      </Attribute>

      <Attribute>
         <Oid>2.40.2.7</Oid>
         <FriendlyName>Position</FriendlyName>
         <AttributeValue DataType="string">
         Administrator
         </AttributeValue>
      </Attribute>
   </Subject>

   <Resource>
      <Oid>1.2.100.1.....</Oid>
      <FriendlyName>Web Service</FriendlyName>
   </Resource>
   <Action>read</Action>
</p1:AuthorizeAccess>

</SOAP:Body>
</SOAP:Envelope>
```

Елементът Subject ще съдържа описание на субекта, който заявява услугата. В общия случай това са данни , които присъстват в издаден от еАвт SAML токен, например: три имена, ЕГН/ЛНЧ, дата на раждане, администрация, заемана длъжност и др. Тези данни са достъпни през Attribute./

Елементът Resource ще съдържа описание на уеб услугата, достъпът до който трябва да бъде оторизиран. Всяка уеб услуга представлява ресурс в БеУ и притежава уникален обектен идентификатор oid. ESB определя обектния идентификатор на основата на данни за уеб услугата.

Елементът Action определя вида на действието, което ще се извършва с ресурса, например: четене (достъп), писане и др.

еОтор изпълнява заявката и връща резултат от следния вид:

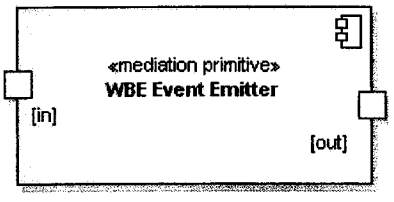
```
<p1:AuthorizationResult xmlns:p1="eauthorization.egov"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
```

```
   <Decision>Permit</Decision>

   <Status>
      <StatusCode>ok</StatusCode>
   </Status>

   <Obligations>
      ...
   </Obligations>
</p1:AuthorizationResult>
```

При неоторизиран достъп (“Deny”) компонентът ще спре изпълнението на заявката, ще регистрира бизнес събитие в Системата за обработка на бизнес събития (през ”Компонент за интеграция със Системата за обработка на бизнес събития” (9.2.4.3)) и ще генерира SOAP Fault с подходящо съдържание.

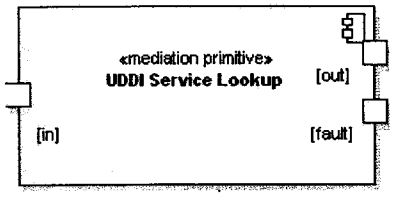
При оторизиран достъп („Permit”) компонентът ще пренасочи заявката към уеб услугата, предоставяна от администрацията.

9.2.4.3 Компонент за интеграция със Системата за обработка на бизнес събития



Това е стандартен компонент от вид mediation primitive, който регистрира бизнес събития в Системата за обработка на бизнес събития. Компонентът е част от IBM ODM.

9.2.4.4 Компонент за интеграция с UDDI



Това е стандартен компонент от вид mediation primitive, който осъществява комуникация с регистър за уеб услуги UDDI и извлича адрес на порт уеб услуга. Този адрес се използва от „Компонент за извикване на уеб услуга” (9.2.4.5) за насочване на SOAP заявката към уеб услуга, предоставяна от администрация.

Компонентът е част от IBM WebSphere ESB.

9.2.4.5 Компонент за извикване на уеб услуга

Това е стандартен компонент от вид mediation primitive, който по адрес на порт на уеб услуга в SMO, пренасочва SOAP заявката към уеб услуга, предоставяна от администрация.

Компонентът е част от IBM WebSphere ESB.

10 МЕТОДИКА ЗА РЕАЛИЗАЦИЯ НА СИСТЕМАТА ЗА ГЕНЕРИРАНЕ И ОБРАБОТКА НА БИЗНЕС СЪБИТИЯ. ПРЕДОСТАВЯНЕ НА ВЪЗМОЖНОСТ ЗА ИНТЕГРАЦИЯ НА СИСТЕМАТА С ДРУГИ ОСНОВНИ КОМПОНЕНТИ ОТ ИНФРАСТРУКТУРАТА НА БЕУ

10.1 Описание

Система за генериране и обработка на бизнес събития (СБС) представлява шина за обработка на бизнес събития, които се генерират от системите в инфраструктурата на БеУ.

Към системата за обработка на събития ще могат да се интегрират основните компоненти в инфраструктурата на БеУ: Портал на БеУ, Валидиращ орган, Системата за електронна оторизация (еОтор), Шината за услуги (ESB) и Журнала на достъпа до ресурси в БеУ.

СБС ще предоставя интерфейс за регистриране на събития. Информационните системи в зона МТИТС ще използват интерфейса на СБС, за да регистрират настъпили в тях събития.

СБС ще поддържа краен набор от събития, като: отказан и разрешен достъп до ресурс, невалиден SAML токен, създаване на нови и промяна на съществуващи политики за достъп, регистриране на нови и промяна на съществуващи ресурси в БеУ и други. По време на изпълнение на проекта при необходимост ще бъдат дефинирани и други събития.

СБС ще позволява на други системи да се абонират за събития и да бъдат известявани при настъпване на абонирано от тях събитие.

СБС ще предостави интерфейс за извличане на информация за абонираните от външни системи събития.

Едно събитие ще има най-малко следните характеристики:

- Уникален номер;
- Точно време на възникване на събитието;
- Вид (номенклатура от идентификатори за вид събитие);
- Обектен идентификатор (oid) на информационна система, където е възникнало събитието;
- Идентификатор на компонент в информационната система, регистрирал събитието;
- Приоритет;

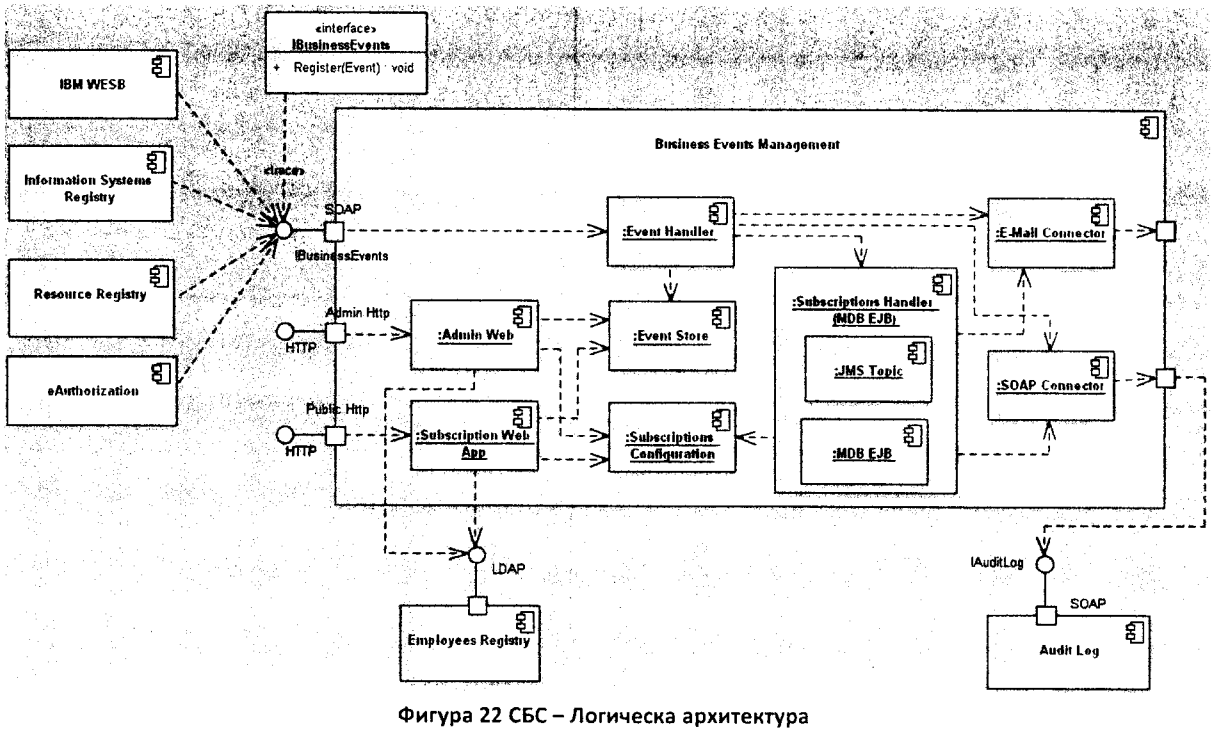
- Секторен псевдоним на физическо лице (когато събитието е свързано с достъп до данни от физическо лице);
- Описание на събитието;
- Данни за събитието.

СБС трябва ще предоставя потребителски интерфейс за справки с възможност за търсене на събития по период на настъпване, приоритет, идентификатор на информационна система и компонент и др.

10.2 Реализация

10.2.1 Компоненти

Логическата архитектура на компонента е изобразена на следващата фигура:



Фигура 22 СБС – Логическа архитектура

10.2.1.1 Уеб услуга за регистриране на събития

Уеб услугата за регистриране на събития е входната точка за системите от БеУ за регистриране на събития. Ще бъде реализирана като SOAP услуга с предоставените средства на IBM ODM.

10.2.1.2 Мениджър обработка на събития

Този компонент представлява ядрото на системата. Тук се извършва логическата обработка на събитията. Компонентът извлича всички правила за обработка на основата на вида регистрирано събитие. В общия случай обработката на събитията е

свързана с тяхното обогатяване с допълнителни данни и пренасочване през конектори към външни системи. Например определен вид събития са свързани с целите на одита в БеУ.

Този вид събития мениджърът обработва регистрираното събитие и изпраща през SOAP конектор към Журнала за достъп до ресурси в БеУ.

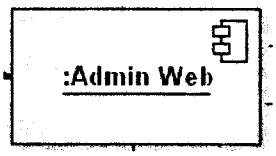
Ще се реализира с предоставените средства на IBM ODM.

10.2.1.3 Хранилище за събития

В хранилището се съхраняват всички събития, за които са дефиниран като „постоянни".

Ще се реализира с предоставените средства на IBM ODM.

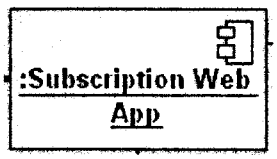
10.2.1.4 Уеб базирано приложение за поддържане на събития



Идентификацията на потребителите ще се осъществява с помощта на потребителско име и парола през Регистър на служителите в ДА. Правата за достъп до функционалността ще се основат на роли.

Ще се реализира с предоставените средства на IBM ODM.

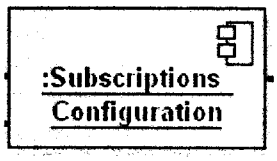
10.2.1.5 Уеб базирано приложение за абониране на събития



Уеб базирано приложение за абониране на събития ще предостави функционалност за абониране за определен вид събития. Например разработчици на електронни услуги в администрациите ще могат да се абонират за събития, свързани със създаване на нови и промяна на съществуващи политики за достъп, регистрите на нови и промяна на съществуващи ресурси в БеУ.

Ще бъде реализирано като JBoss SEAM приложение с JBoss RichFaces потребителски интерфейс.

10.2.1.6 Конфигурация на абонаменти

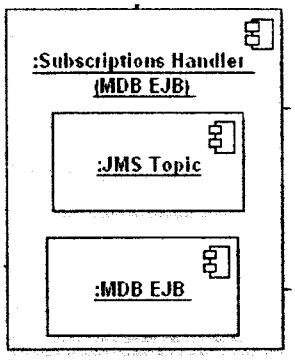


Този компонент е отговорен за поддържане на конфигурацията за абонаменти.

Конфигурацията определя връзката между вид събитие, заинтересована ИС и канал за известяване.: електронна поща или SOAP уеб услуга.

За съхранението на конфигурацията ще се използва релационна СУБД.

10.2.1.7 Мениджър абонаменти за събития



Този компонент е отговорен за управление на процеса за обработка на абонаменти за събития.

Състои се от JMS Topic и Message Driven EJB

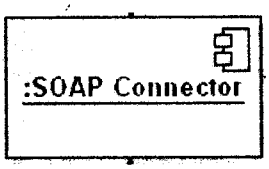
„Мениджърът обработка на събития" регистрира съобщения в опашката JMS Topic за събития, за които има регистрирани абонаменти.

Специализиран компонент, реализиран като Java MDB, получава известие за влезли в опашката съобщения, обработва ги и на основата на данни за информационните системи изпраща известия на предварително дефиниран канал за връзка: като електронна поща или като SOAP съобщение. Във втория случай фирма Бул Ес Ай ще дефинира и специфицира интерфейс, който ИС трябва да реализират, за да могат да бъдат известявани чрез уеб услуга.

10.2.1.8 Компонент за интеграция със SMTP сървър

Този компонент е отговорен за осъществяване на връзка с SMTP сървър и изпращане на съобщения. IBM WebSphere Application Sever предоставя SMTP конектор.

10.2.1.9 Компонент за интеграция с SOAP уеб услуги



Този компонент е отговорен за осъществяване на обмен на данни с SOAP уеб услуги. IBM WebSphere Application Sever предоставя SOAP конектор.

11 МЕТОДИКА ЗА РЕАЛИЗАЦИЯ НА ЖУРНАЛ НА ДОСТЪПА ДО РЕСУРСИ В БЕУ, ОСНОВАВАЩ СЕ НА СИСТЕМАТА ЗА ГЕНЕРИРАНЕ И ОБРАБОТКА НА БИЗНЕС СЪБИТИЯ

11.1 Описание

Журналът на действията в системата ще съхранява история на достъпа до ресурси и извършените действия с ресурсите.

Журналът ще бъде интегриран със Системата за генериране и обработка на събития (СБС) . Всяко регистрирано събитие в СБС има най-малко следните характеристики:

- Уникален номер;
- Точно време на възникване на събитието;
- Вид (номенклатура от идентификатори за вид събитие);
- Обектен идентификатор (oid) на информационна система, където е възникнало събитието;
- Идентификатор на компонент в информационната система, регистрирал събитието;
- Приоритет;
- Описание на събитието;
- Секторен псевдоним (когато събитието представлява е свързано с достъп до данни от физическо лице);
- Данни за събитието.

На основата на реализирания от СБС абонаментен принцип, журналът ще получава информация за регистрираните от останалите системи събитията и ще ги записва по подходящ начин. Първоначално Журналът ще бъде абониран само за събития, свързани с достъп до лични данни. Тези събития, както и регистриращите ги информационни системи, ще бъдат дефинирани от Възложителя по време на първия етап от изпълнението на дейността. Данните за достъп не трябва да съдържат ЕГН или ЛНЧ, а само секторен псевдоним на физическото лице.

С помощта на журнала физически лица ще могат да проверяват за извършен достъп до личните им данни. Информацията, до която физически лица ще имат достъп, ще включва най-малко:

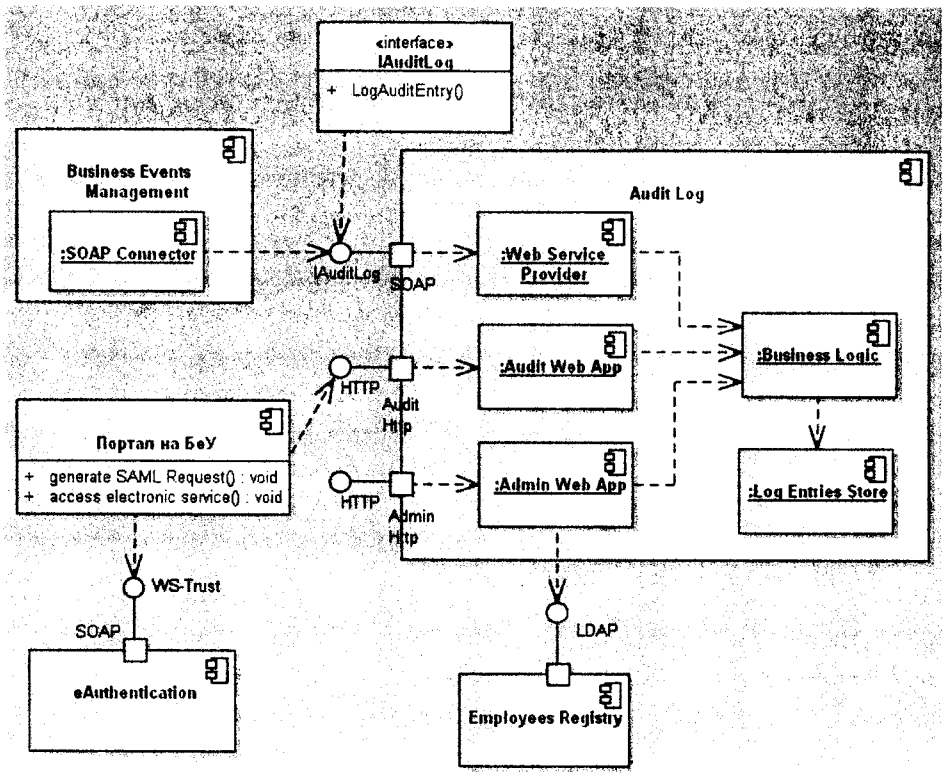
- точно време на настъпване на събитието (осъществен достъп до лични данни);
- вид на данните, до които е бил осъществен достъп;
- администрация, заявила достъп до данните;
- служител в администрацията, заявил данните;

- администрация, предоставила данните:
- (при технологична възможност) служител в администрацията, предоставил данните.

Достъпът до тази информация ще се осъществява през специализирано приложение и ще изисква идентификация на потребител с носител на електронна идентичност. Тъй като все още няма реално функциониращ Валидиращ орган и съответно електронна идентичност, то в обхвата на тази дейност е само изграждане на функциониращ прототип на журнал.

11.2 Реализация

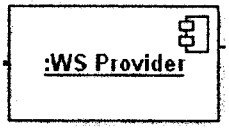
Логическата структура на ЖР е изобразена на следващата диаграма:



Фигура 23 ЖР – Логическа архитектура

11.2.1 Компоненти

11.2.1.1 Уеб услуга за достъп до журнала



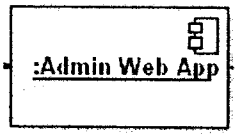
Компонентът ще предостави уеб услуга (SOAP/HTTPS) за достъп до справочника от други информационни системи.

SOAP уеб услуга ще реализира метод LogAuditEntry с параметър

На основата на реализирания от СБС абонаментен принцип, журналът ще получава информация за регистрираните от останалите системи събитията и ще ги записва по подходящ начин. Първоначално Журналът ще бъде абониран само за събития, свързани с достъп до лични данни. Тези събития, както и регистриращите ги информационни

системи, ще бъдат дефинирани от Възложителя по време на първия етап от изпълнението на дейността. Данните за достъп не трябва да съдържат ЕГН или ЛНЧ, а само секторен псевдоним на физическото лице.

11.2.1.2 Уеб базирано приложение за администриране на журнала

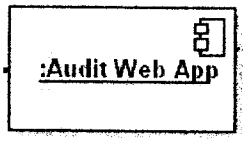


Уеб базирано приложение, предоставящо функционалност за администриране на Журнала.

Идентификацията на потребителите ще се осъществява с помощта на потребителско име и парола през Регистър на служителите в ДА. Правата за достъп до функционалността ще се основат на роли.

Ще бъде реализирано като JBoss SEAM приложение с JBoss RichFaces потребителски интерфейс.

11.2.1.3 Уеб базирано приложение за следене на достъпа



Уеб базирано приложение за абониране на събития ще предостави функционалност за абониране за определен вид събития. Ще бъде достъпно за граждани. Идентификацията ще се основава на еИд и SAML токен. Приложението ще бъде достъпно през ПБеУ, който ще инициира идентификацията на потребителя с еИд.

ПБеУ ще получи SAML атестат от Валидиращия орган, ще генерира WS-Trust заявка към eAwt.

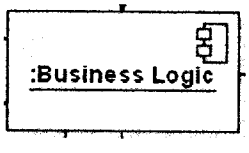
Сценарият е описан подробно в „Автентикация на физическо лице -гражданин или служител в ДА с еИД" (7.2.2.2).

Получавайки SAML токен от еАвт, ПБеУ ще кодира токена в HTTP header параметър SAMLAssertion и ще изпрати HTTP заявката към приложението.

Приложението от своя страна извлича SAML токена от HTTP заявката, опционално валидира SAML токена през еАвт, създава потребителска сесия и изобразява екран за търсене на регистрирани журнални събития. Търсенето ще бъде по разнообразни критерии, като: период на настъпване на събитието, информационна система, вид данни, до които е бил осъществен достъпът и др.

Ще бъде реализирано като JBoss SEAM приложение с JBoss RichFaces потребителски интерфейс.

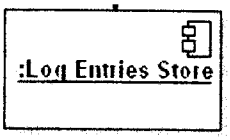
11.2.1.4 Бизнес логика



Компонентът ще реализира логика за търсене и извличане на записи от журнала.

Ще бъде реализирано като набор от Java EJB компоненти.

11.2.1.5 Хранилище на журнални записи

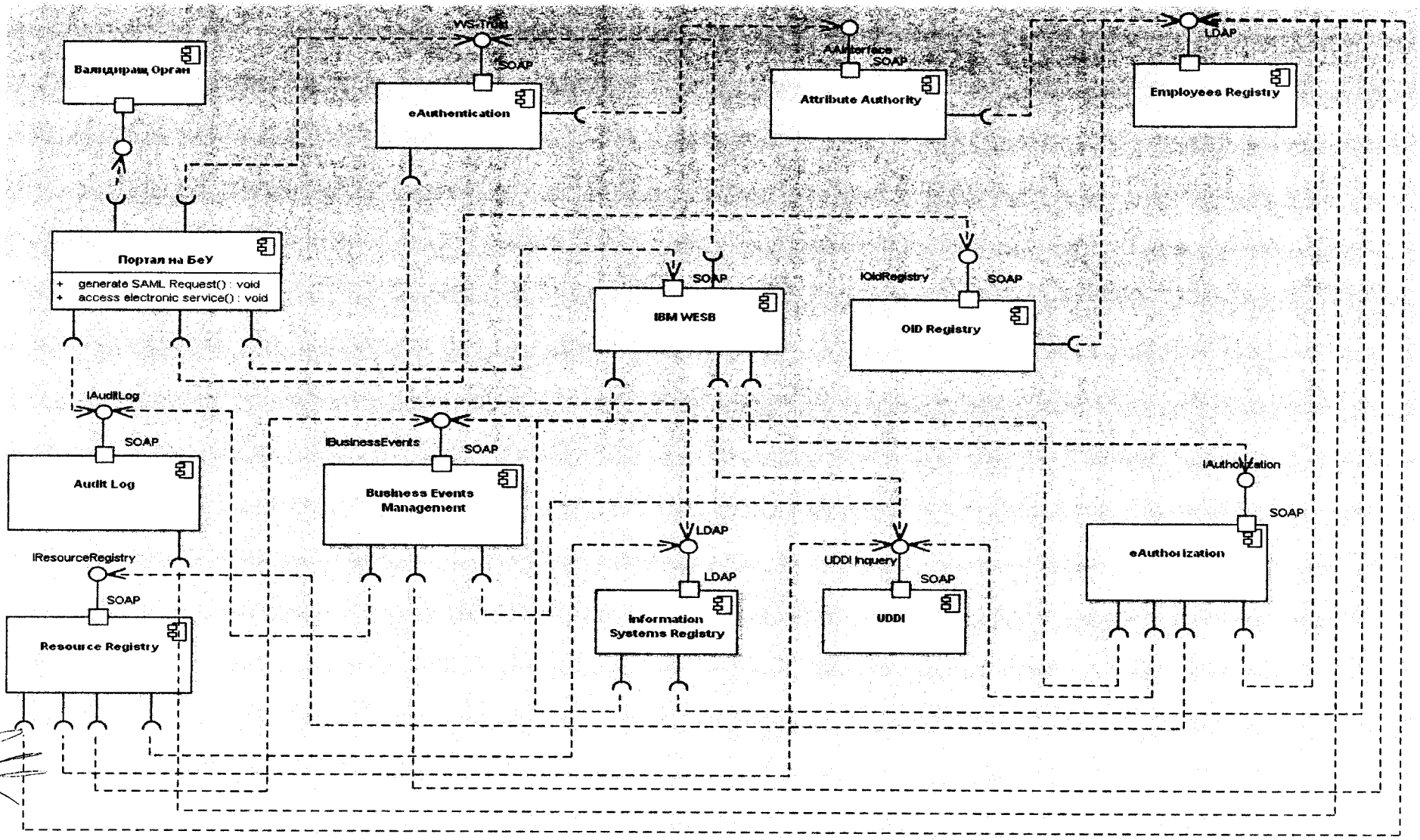


Хранилището съхранява журналните записи. Ще бъде реализирано като набор от таблици, изгледи и други дизайн елементи в релационна СУБД

12 МЕТОДИКА ЗА ИНТЕГРАЦИЯ НА РАЗРАБОТЕНИТЕ КОМПОНЕНТИ И СИСТЕМИ В СРЕДАТА НА БЕУ

12.1 Комуникации между отделните компоненти

Връзката между отделните компоненти е изобразена следващата фигура:


Валидиращ Орган
WS-Trust
SOAP
eAuthentication
AAInterface
SOAP
Attribute Authority
LDAP
Employees Registry
Портал на БеУ
+ generate SAML Request() : void
+ access electronic service() : void
SOAP
IBM WESB
IOidRegistry
SOAP
OID Registry
IAuditLog
SOAP
Audit Log
IBusinessEvents
SOAP
Business Events Management
IAuthorization
SOAP
eAuthorization
IResourceRegistry
SOAP
Resource Registry
LDAP
LDAP
Information Systems Registry
UDDI Inquery
SOAP
UDDI

12.1.1 Портал на БеУ

Порталът на БеУ обменя данни със следните компоненти:

Компонент	*Използвана функционалност*	*Протокол*
Компонент за еднократна автентикация	Заявка за издаване на SAML токен	WS-Trust / SOAP / HTTPS
Журнал на достъпа до ресурси в БеУ	Пренасочване към Уеб базирано приложение за следене на достъпа след идентификация на потребител с еИд през Валидиращ Орган	HTTPS
Регистър на обектните идентификатори в БеУ	Изобразяване на дървото на обектните идентификатори в портлет	SOAP / HTTPS
ESB	Пренасочване на SOAP заявки за уеб услуги	SOAP / HTTPS
Регистър на уеб услуги (UDDI)	Извличане на адрес на уеб услуги, предоставяни от компонентите в инфраструктурата на БеУ	UDDI / SOAP / HTTPS

12.1.2 Компонент за еднократна автентикация (еАвт)

еАвт обменя данни със следните компоненти:

Компонент	*Използвана функционалност*	*Протокол*
Справочник за атрибути	Извличане на допълнителна информация за служител	SOAP / HTTPS
Регистър на информационните системи	Автентикация на информационна система, за която се заявява издаване на SAML токен.	LDAP с SSL

12.1.3 Справочник за атрибути

Справочникът за атрибути обменя данни със следните компоненти:

Компонент	*Използвана функционалност*	*Протокол*
Регистър на служителите в ДА	Извличане на допълнителни данни за служител в процеса на автентикация (издаване на SAML токен)	LDAP с SSL
Регистър на служителите в ДА	Идентификация на потребител Уеб приложение за управление на справочника	LDAP с SSL
Регистър на информационните системи	Извличане на допълнителни данни за информационна система в процеса на автентикация (издаване на SAML токен)	LDAP с SSL

12.1.4 Регистър на ресурсите в ДА

Регистърът на ресурсите в ДА обменя данни със следните компоненти:

Компонент	*Използвана функционалност*	*Протокол*
Регистър на служителите в ДА	Идентификация на потребителите в Уеб базирано приложение за поддържане на ресурсите	LDAP с SSL
Регистър на уеб услуги (UDDI)	Извличане, промяна и създаване на записи	UDDI /SOAP / HTTPS
Система за обработка на бизнес събития	Регистриране на събитие при промяна в запсиисте в регистъра	SOAP / HTTPS
Регистър на информационните системи	Извличане, промяна и създаване на записи	LDAP с SSL

12.1.5 Регистър на информационните сиситеми

обменя данни със следните компоненти:

Компонент	*Използвана функционалност*	*Протокол*
Регистър на служителите в ДА	Идентификация на потребителите в Уеб базирано приложение за поддържане на записите в регистъра	LDAP с SSL
Регистър на уеб услуги (UDDI)	Извлича адреса на уеб услугата на	UDDI /SOAP / HTTPS
Система за обработка на бизнес събития	Регистриране на събитие при промяна на записи в регистъра	SOAP / HTTPS

12.1.6 Регистър на обектните идентификатори

обменя данни със следните компоненти:

Компонент	*Данни*	*Протокол*
Регистър на служителите в ДА	Идентификация на служител за работа с приложението за администрариране	LDAP с SSL

12.1.7 ESB

Компонент	*Използвана функционалност*	*Протокол*
Компонент за еднократна автентикация (еАвт)	Валидиране на SAML токен	WS-Trust / SOAP / HTTPS
Регистър на уеб услуги (UDDI)	Извлича адресите на уеб услуги на компоненти в инфраструктурата на БеУ Извлича адресите на уеб услуги, предоставяни от информационни системи в администрациите	UDDI /SOAP / HTTPS

Система за обработка на бизнес събития	Регистриране на събитие – невалиден SAML токен Регистриране на събитие - отказан достъп до ресурс	SOAP / HTTPS
Компонент за електронна оторизация (еОтор)	Оторизиране на достъп до уеб услуга	SOAP / HTTPS

12.1.8 Компонент за електронна оторизация

Компонент за електронна оторизация (еОтор) обменя данни със следните компоненти:

Компонент	*Използвана функционалност*	*Протокол*
Регистър на уеб услуги (UDDI)	Извлича адреса на уеб услугата на Регистъра на ресурси в ДА	UDDI /SOAP / HTTPS
Система за обработка на бизнес събития	Регистрира събитие при промяна на политика за достъп	SOAP / HTTPS
Регистър на ресурсите	Извлича данни за ресурсите, които се използват при разработка на политиките за достъп	SOAP / HTTPS
Регистър на служителите в ДА	Идентификация на потребител в Уеб базирано приложение за поддържане на политиките за достъп	LDAP с SSL

12.1.9 Система за обработка на бизнес събития

обменя данни със следните компоненти:

Компонент	*Използвана функционалност*	*Протокол*
Регистър на уеб услуги (UDDI)	Извлича адреса на уеб услугата на	UDDI /SOAP / HTTPS
Журнал на достъпа до ресурси в БеУ	Регистрира събитие от вид «достъп до ресурс» на основата на абонамент	SOAP / HTTPS
Регистър на служителите в ДА	Идентификация на потребители в Уеб базирано приложение за поддържане на събития	LDAP с SSL

12.1.10 Журнал на достъпа до ресурси в БеУ

обменя данни със следните компоненти:

Компонент	*Използвана функционалност*	*Протокол*
Регистър на служителите в ДА	Автентикация на потребители-служители в ДА в Уеб базирано приложение за администриране на журнала	LDAP с SSL

13 ИЗПОЛЗВАНА МЕТОДИКА ЗА РЕАЛИЗАЦИЯТА НА КОМПОНЕНТИТЕ

Методиката за реализация включва „напасване" на отделните дисциплини на избраната методология за разработка на приложения RUP към специфичните изисквания за разработка на отделните компоненти.

Фирма Бул Ес Ай има разработен обобщен SPEM (Software and Systems Process Engineering Meta-Model) модел на методика за разработка, основаващ се на методологията RUP. SPEM е UML профил, който представлява мета-модел и концептуална рамка, осигуряваща необходимите концепции за моделиране, документиране, представяне, управление и взаимозаменяемост на дейностите по разработка на софтуерни приложения.

- Основни предимства при използване на SPEM мета-модели:
- Осигуряване на стандартизирани и конфигурируеми библиотеки от процес, дейности, артефакти, роли и зависимости между всички тях за лесно преизползване;
- Създава основа за поддръжка, управление и еволюиране на процесите на софтуерна разработка;
- Тъй като SPEM се основава на UML, то такъв модел може да се разглежда като "независим от платформата модел" (PIM) според подхода Model-driven Architecture (MDD) и съответно е подходящ за трансформиране. Фирма Бул Ес Ай използва разработена трансформация: от SPEM към JIRA (дейностите в SPEM се регистрират като дейности в JIRA).

Всички дейности, описани по-нататък в тази глава, са според обобщения SPEM мета-модел, който Бул Ес Ай ще конфигурира според спецификата на реализация на всяка една от системите в този проект.

13.1 Фаза Проектиране

Целта на фаза „Проектиране" е създаване на:
- Дизайн на бизнес процесите;
- Реализация на случаите на употреба;
- Логически и физически модел на данните;
- Техническа (софтуерна и инфраструктурна) архитектура на системата.

Фаза „Проектиране" използва резултатите от фаза „Планиране", а именно:
- Модел на бизнес процесите на администрацията;
- Домейн модел на администрация;
- Модел на случаите на употреба.

Основни роли във фазата изпълняват:

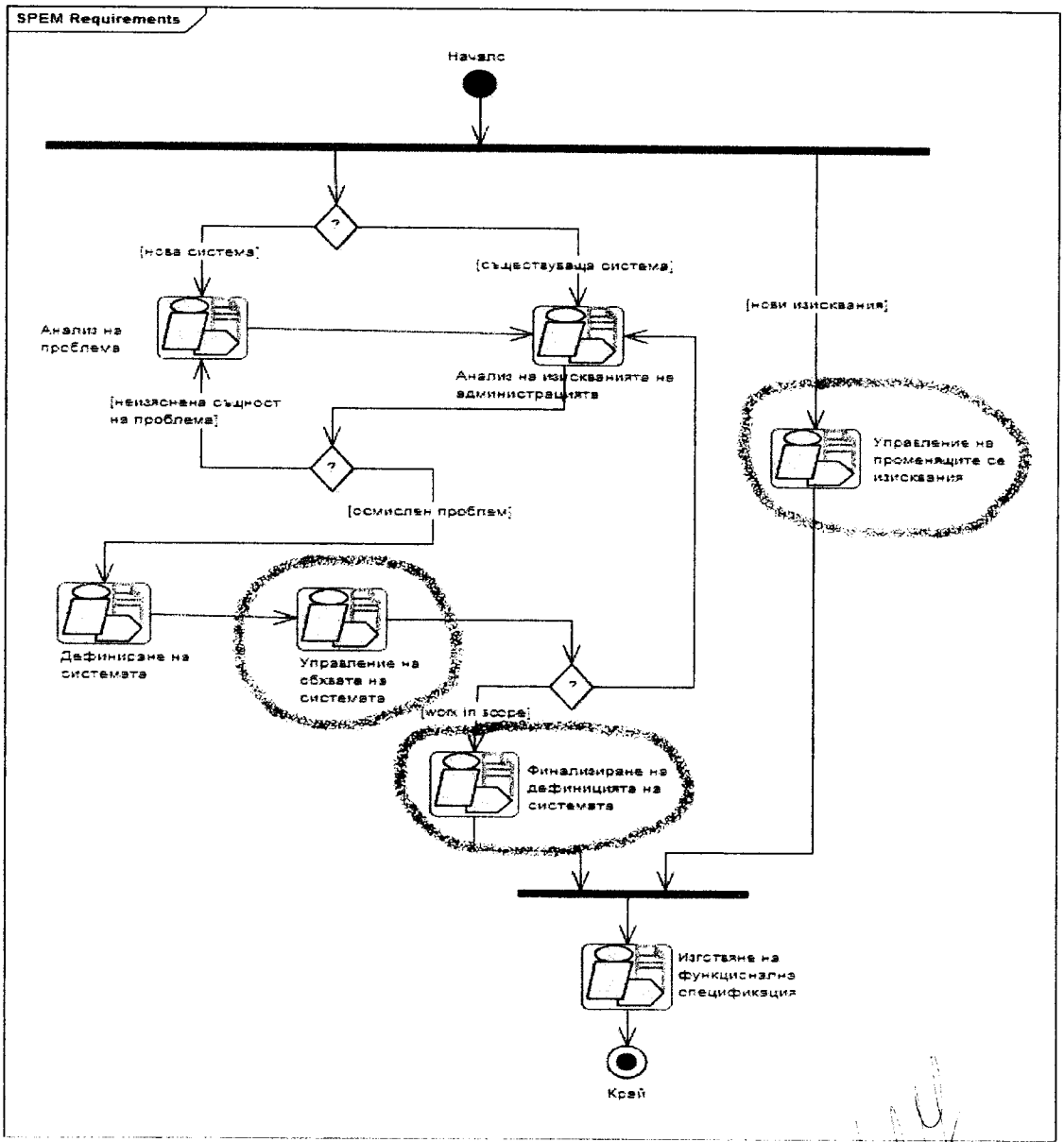
- Софтуерен архитект;
- Дизайнер;
- Дизайнер на бази данни.

Резултати от изпълнението на фазата са следните артефакти:

- BPMN2 модел (формат: BPMN2 диаграми в Sparx EA);
- Модел на услугите (формат: BPMN2 диаграми в Sparx EA);
- Компонентен модел на ниво интерфейси (формат: UML class диаграми в Sparx EA);
- Логически модел данни (формат: UML class диаграми в Sparx EA);
- Физически модел данни(формат: UML class диаграми в Sparx EA).

13.1.1 Дейности от дисциплина „Изисквания"

Основните дейности от дисциплина „Изисквания", които са застъпени в етап „Проектиране", са обобщени на следващата диаграма:



фигура 24

Описание на дейностите:

Дейност: Управление на променящите се изисквания	
	Отговорна роля: Системен анализатор
	Вход: Изискване за промяна (Change Request) Изисквания на администрацията Модел на начините на използване (Use Case Model) Компонентен Модел
	Стъпки: Структуриране на модела на начините на използване Дефиниране на Include-връзка между Use Cases Дефиниране на Extend-Relationships между Use Cases Дефиниране на Generalizations между Use Cases Дефиниране на Generalizations между актьори Управление на зависимостите между артефакти Присвояване на атрибути на артефактите (сложност, риск, приоритет и др.) Реализиране и проверка за проследяемост (Traceability) Управление на променящите се изисквания
	Резултат: Модела на начините на използване (актуализиран)

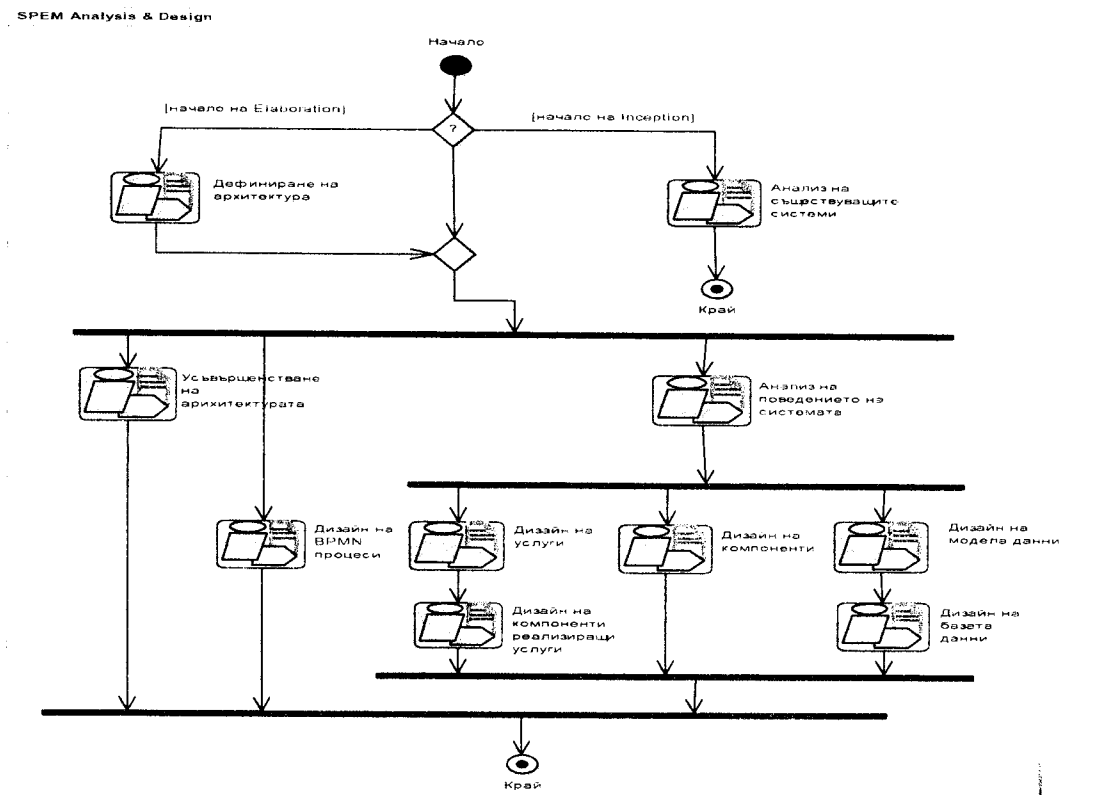
Дейност: Управление на обхвата на системата	
	Отговорна роля: Системен анализатор
	Вход: Модел на начините на използване
	Стъпки: Приоритизиране на начините на използване Дефиниране на подмножество от сценарии, които представляват важна част от функционалността на системата. Дефиниране на архитектурните решения
	Резултат:

	Модел на начините на използване Архитектурни решения

Дейност: Финализиране на дефиницията на системата	
	Отговорна роля: Бизнес анализатор Дизайнер на потребителски интерфейс
	Вход: Модел на начините на използване
	Стъпки: Детайлизиране на софтуерните изисквания Детайлизиране на Use Case Моделиране на потребителския интерфейс Създаване на прототип на потребителския интерфейс
	Резултат: Модел на начините на използване (подобрен) Модел на потребителския интерфейс (storyboard) Прототип на потребителския интерфейс

13.1.2 Дейности от дисциплина „Анализ и дизайн"

Основните дейности от дисциплина „Анализ и дизайн", които са застъпени във фаза „Проектиране", са обобщени на следващата диаграма:



фигура 25

Описание на дейностите:

Дейност: Дефиниране на архитектурата	
	Отговорна роля: Софтуерен архитект
	Вход: • Референтна архитектура на системата • Модел на начините на използване на системата • Нефункционални изисквания • Архитектурни и дизайн шаблони
	Стъпки: 1. Архитектурен анализ • Разработка на преглед на архитектурата; • Описание на наличните активи; • Дефиниране на под-системите на високо ниво; • Разработка на deployment model; • Идентифициране на механизъма за извършване на анализ; • Създаване на реализации на начините на използване(Use-Case Realizations) • Обсъждане на резултатите. 2. Анализ на начин на използване (Use Case) • Допълване на описанието начин на използване; • За всяка реализации на начин на използване -идентифициране на анализ класовете от описанието на начин на използване -разпределение на функционалност по анализ класове • За всеки идентифициран анализ клас -Описание на отговорностите -Описание на атрибути и асоциации -Дефиниране на атрибути -Дефиниране на асоциации между анализ класовете -Описание на зависимостите от събития
	Резултат: • Deployment model (UML Deployment диаграма)

	• Модел на реализациите на начините на използване (актуализиран)

Дейност: Анализ на съществуващите системи	
	Отговорна роля: Софтуерен архитект, Дизайнер
	Вход: Документация от предишни проекти
	Стъпки: 1. Провеждане на периодични срещи с представители на администрациите за изясняване на текущото състояние 2. Анализ на използваните технологии 3. Анализ на използваните операционни системи 4. Анализ на текущата инфраструктура
	Резултат: • Описание на съществуващите системи в МТИТС

Дейност: Дизайн на BPMN процеси	
	Отговорна роля: Дизайнер на бизнес процеси
	Вход: • Модел на бизнес use case реализации • Модел на бизнес процесите (To-Be) • Домейн модел
	Стъпки: 1. Анализ на бизнес обектите 2. Анализ на изискванията за автоматизация, съдържащи се в Модел на бизнес процесите 3. Анализ на Модел на бизнес use case реализации 4. Създаване на BPMN2 процеси
	Резултат: • BPMN2 диаграми

Дейност: Анализ на поведението на системата	
	Отговорна роля: Дизайнер

Вход:

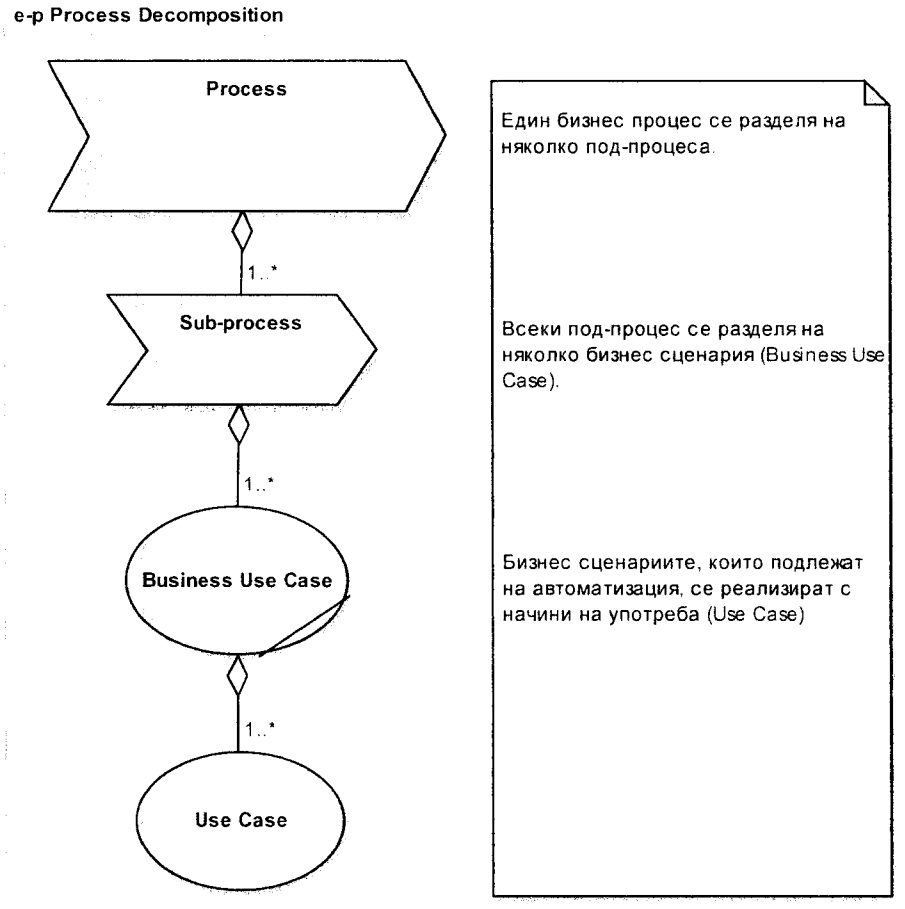
Модел Бизнес процеси (To-Be), Бизнес Use Case модел, Домейн Модел

Стъпки:

1. Декомпозиция на домейна

- Идентификация на услуги чрез декомпозиция на бизнес процеси

  Разделяне на бизнес процес на под-процеси, всеки от който се реализира с бизнес сценарии, а бизнес сценариите се реализират с начини на използване (Use Case):



- Идентификация на услуги чрез анализ на вариациите на системата

  Анализът на вариациите на системата създава дизайн, който е нечувствителен на промени. По време на анализа се установяват взаимозависимости, като например:

  Йерархии на видовете

  Ход на процеси (вариации в изпълнението на процес)

  Бизнес правила (вариации в бизнес правилата)

  Анализът на вариациите може да доведе до идентифициране на услуги.

2. Анализ на целите на администрацията

- Идентификация на цели и подцели

  Декомпозиция на бизнес цели в конкретни подцели

- Идентификация на услуги за постигане на подцелите

  За постигане на определена цел може да се дефинират услуги.

- Идентификация на индикатори KPI и дефиниране на измерими показатели

  За всяка подцел се идентифицират KPI и измерими показатели за следене на изпълнението на целта, например: степен на удовлетвореност от предоставяне на услуги, следене на продължителност на изпълнение на услуги, време за изпълнение на определена дейност и др.

3. Анализ на съществуващите системи

- Идентификация на съществуващите системи, с които ще се обменят данни

  Анализира се функционалността на съществуващите системи и се търсят общи функционалности, които могат да бъдат реализирани като услуги

- Методика за реализация на услугите

  Определяне на компоненти и технологии за реализация: адаптери, Java Connecticity Architecture (JCA)или др.

4. Анализ на начините на използване (Use Case Analysis)

Допълва описанието на начините на използване

- За всяка реализация на начините на използване (Use Case Realization):

  -Идентифициране на анализ класовете (boundary-control-entity)

  -Разпределяне на отговорностите между анализ класовете

- За всеки дефиниран анализ клас (boundary-control-entity):

  -Описание на отговорностите

  -Описание на атрибути и асоциации

  -Дефиниране на атрибути

- Дефиниране на връзките между анализ класовете (boundary-control-entity)
- Описание на последователността на обработка на събития от анализ класовете (boundary-control-entity)

	5. Идентифициране на подсистеми и компоненти • Идентифициране на подсистеми • Идентифициране и специфициране на интерфейси на подсистеми 6. Идентифициране на дизайн елементи • Идентифициране и специфициране на събития • Идентифициране и специфициране на сигнали 7. Идентифициране на класове
	Резултат: • Модел на услугите (съдържа само портфолио на услугите и интерфейси) • Анализ модел (Analysis model)

Дейност: Дизайн на услуги	
	Отговорна роля: Дизайнер
	Вход: Нефункционални изисквания Модел на бизнес процесите (To-Be) Домейн модел Модел на услугите
	Стъпки: 1. Идентификация на зависимостите на услугите една от друга Определяне на зависимостите при извикване на услугата, например една услуга може да изисква други услуги да бъдат приключили преди нейното изпълнение или да се изпълняват по време на нейното изпълнение. 2. Идентификация на оркестрацията на услугите и техния ход Преценка на необходимостта от създаване на услуги на по-горно ниво (композитни услуги), които реализират логиката на последователност на изпълнение 3. Идентификация на политики за сигурност и други нефункционални изисквания, които гарантират качество на услугите (QoS)

	4. Спецификация на SOAP съобщенията
	Резултат: Модел на услугите за различните системи (пълен)

Дейност: Дизайн на компоненти, реализиращи услуги	
	Отговорна роля: Дизайнер
	Вход: Модел услуги (резултат от дейности „Анализ на поведението на системата" и „Дизайн на услуги") Референтна архитектура на администрация
	Стъпки: 1. Разпределение на услуги към компоненти Услугите се реализират от компоненти. Множество компоненти са дефинирани в Референтната архитектура на администрацията. След анализ на функционалността, предлагана от услугата, се подбират онези компоненти, които са най-подходящи за реализация на функционалността. 2. Разпределение на компоненти към слоеве Компонентите принадлежат на различни слоеве. Тук отново се използва Референтната архитектура на администрацията
	Резултат: Компонентен модел на системата (описани компоненти, реализиращи услуги)

Дейност: Дизайн на компоненти	
	Отговорна роля: Дизайнер
	Вход: Модел услуги (резултат от дейности „Анализ на поведението на системата")
	Стъпки: 1. Дизайн на начините на използване 2. Дизайн на подсистемите 3. Дизайн на класове 4. Дизайн на тест класове и пакети
	Резултат:

	• Компонентен модел (описани всички компоненти) за системата

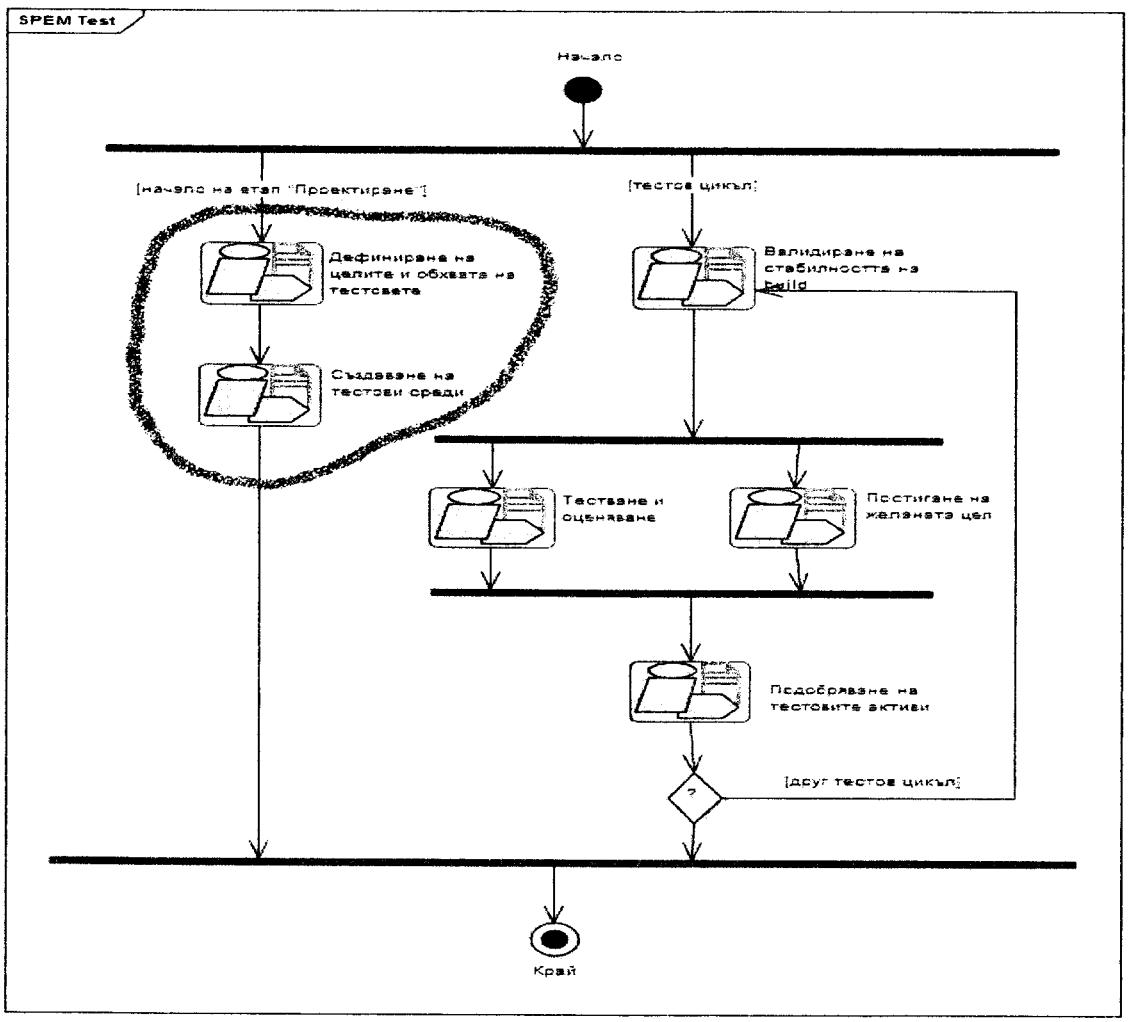
Дейност: Дизайн на модела данни	
	Отговорна роля: Дизайнер на бази данни
	Вход: Домейн Модел (от дейност „Разработка на домейн модел"/дисциплина „Бизнес моделиране") Анализ модел (analysis model)
	Стъпки: 1. За всеки анализ обект от вид entity се определя дали да бъде записван в база данни 2. Дефиниране на атрибути на entity (тип размер) 3. Дефиниране на primary key 4. Създаване на Entity-relationship диаграми
	Резултат: • Модел връзка-отговорност (E/R модел)

Дейност: Дизайн на базата данни	
	Отговорна роля: Дизайнер на бази данни
	Вход: Модел Връзка-Отговорност (E/R model) от Дейност „Дизайн на модел данни"
	Стъпки: 1. Разпределение на обекти от E/R модел към таблици в базата данни Ще се използва предоставената от Sparx EA трансформация от E/R към физически модел данни. 2. Оптимизиране на модела данни за производителност 3. Оптимизиране на достъпа до данни (използване на ORM като JPA/Hibernate) 4. Дефиниране на характеристиките на сървър база данни (например: размер на tablespaces) 5. Дефиниране на таблиците с номенклатури 6. Дефиниране на правилата за гарантиране на интегритет на данните и връзките (foreign key и check ограничения)

	Резултат: • Спецификация на физическия модел данни • DDL скриптове за създаване на таблица, изгледи, ограничения

13.1.3 Дейности от дисциплина „Тестване"

Основните дейности от дисциплина „Тестване", които са застъпени във фаза „Проектиране", са обобщени на следващата диаграма:



фигура 26

Описание на дейностите:

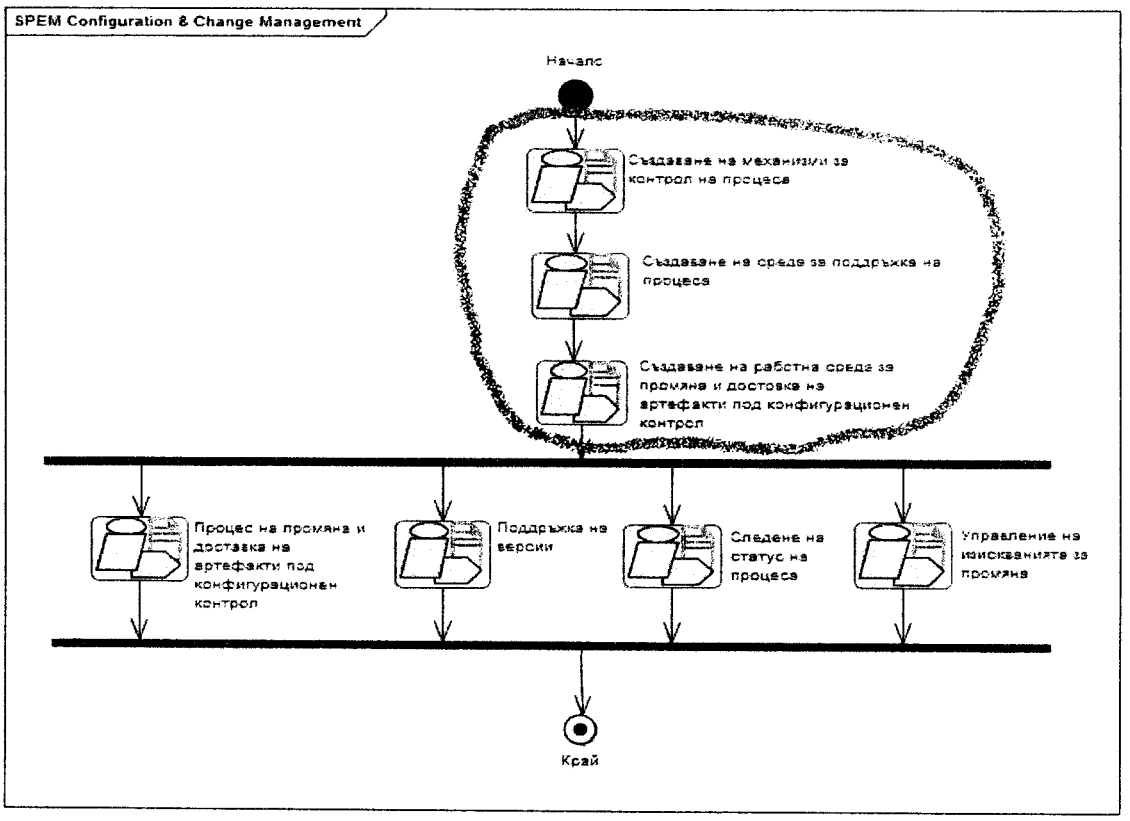
Дейност: Дефиниране на целите и обхвата на тестовете	
	Отговорна роля: Тест дизайнер, Софтуерен архитект
	Вход: • Референтна архитектура на системата

	• Компонентен модел • Модел на начините на използване (Use Case Model) • Модел на услугите • BPMN2 модел
	Стъпки: 1. Идентифициране на стратегия за използване на наличните ресурси (хардуер, хора) 2. Дефиниране на оптимален обхват за тестовете 3. Формално описание на процедурите за тестване (артефакти, дейности, отговорни роли) 4. Дефиниране на механизъм за оценка на тестовите резултати и генериране на справки.
	Резултат: • План за провеждане на тестове на системата

Дейност: Създаване на тестови среди	
	Отговорна роля: Тест дизайнер, Софтуерен архитект
	Вход: • План за провеждане на тестове • Референтна архитектура • Модел на начините на използване (Use Case Model) • Модел на услугите • BPMN2 модел
	Стъпки: 1. Създаване на тестова среда за тестване на BPMN2 процеси 2. Създаване на тестова среда за тестване на Java приложенията 3. Създаване на тестова среда за тестване на интеграционни процеси
	Резултат: Тестова среда за тестване на BPMN2 процеси Тестова среда за приложенията Създаване на тестова среда за тестване на интеграционни процеси

13.1.4 Дейности от дисциплина „Конфигурация и управление на промените"

Основните дейности от дисциплина „Конфигурация и управление на промените", които са застъпени във фаза „Проектиране", са обобщени на следващата диаграма:



фигура 27

Описание на дейностите:

Дейност: Създаване на механизми за контрол на процеса	
	Отговорна роля: Ръководител на CM процес
	Стъпки: 1. Създаване на процеса (контрол на версиите, етикетиране/labeling) 2. Дефиниране на workflow 3. Дефиниране на статуси на задачи (например: new, assigned, finished, closed) 4. Дефиниране на протокол за известяване при настъпване на събития ()
	Резултат: 1. План за управление на конфигурацията и управление на промените

Дейност: Създаване на среда за поддръжка на процеса

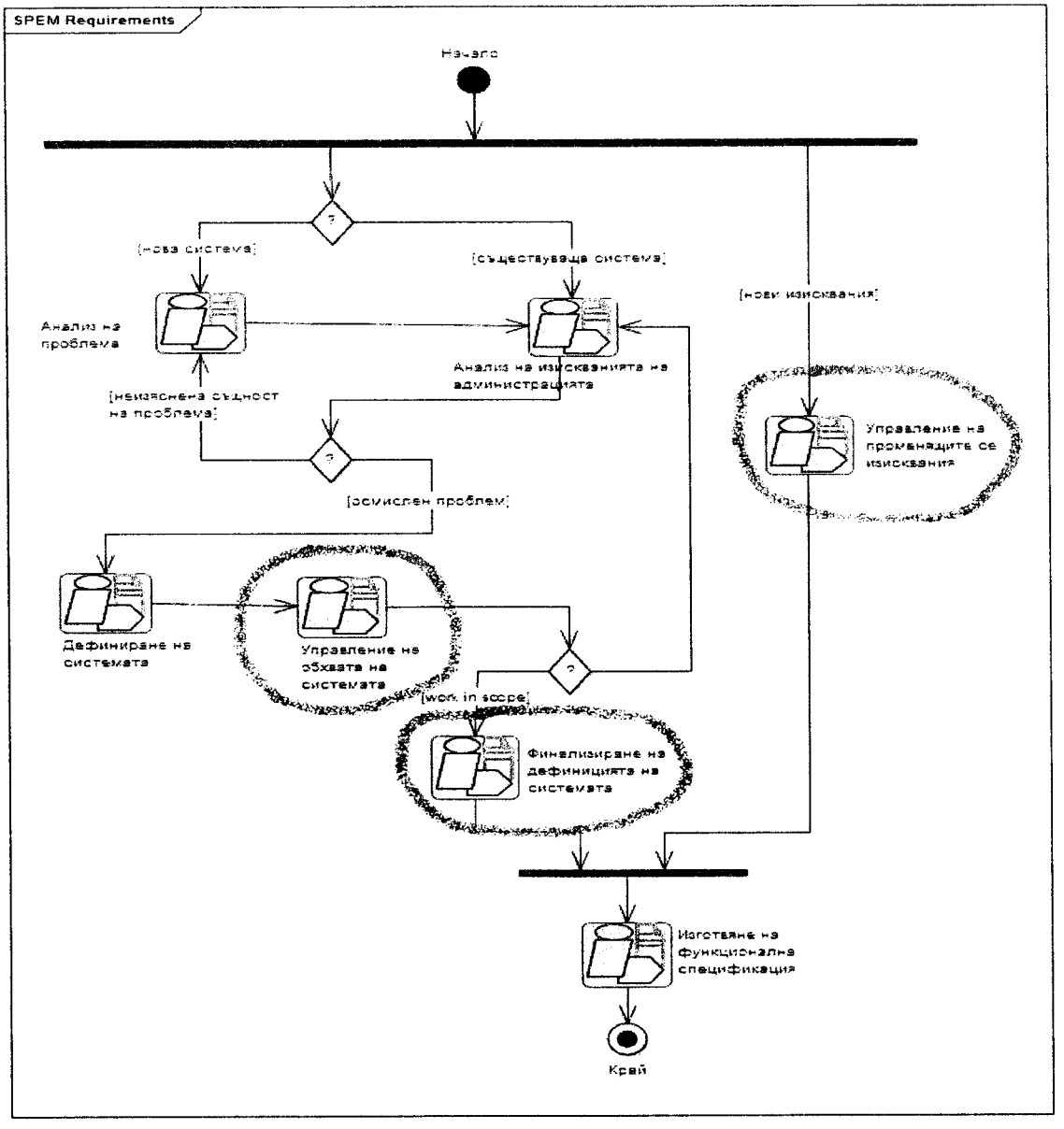
	Отговорна роля: Софтуерен архитект
	Вход: 2. План за управление на конфигурацията и управление на промените
	Стъпки: 1. Инсталиране на JIRA и CVS (Concurrent Versions System) 2. Дефиниране на статуси на задачи (например: new, assigned, finished, closed) 3. Дефиниране на workflow и отговорни роли 4. Създаване на проекти в JIRA за отделните модули 5. Създаване на CVS (svn) хранилище
	Резултат: 3. СМ среда за МТИТС

Дейност: Създаване на работна среда за промяна и доставка на артефакти под конфигурационен контрол	
	Отговорна роля: Софтуерен архитект
	Вход: Описание на СМ среда
	Стъпки: Инсталиране на работната станция на разработчик на необходим клиент за достъп до JIRA и CVS (subversion svn,) Настройка на средата на разработка eclipse за работа с Subversion svn Настройка на средата на разработка Microsoft Visusla Studio за работа с CVS
	Резултат: Настроени среди за разработка (development environment) за участие в СМ процеса

13.2 Фаза Разработка

13.2.1 Дейности от дисциплина „Изисквания"

Основните дейности от дисциплина „Изисквания", които са застъпени във фаза „Разработка", са обобщени на следващата диаграма:



фигура 28

Описание на дейностите:

Дейност: Управление на променящите се изисквания	
	Отговорна роля: Системен анализатор
	Вход: • Изискване за промяна (Change Request) • Изисквания на администрацията • Модел на начините на използване (Use Case Model) • Компонентен Модел
	Стъпки: 1. Структуриране на модела на начините на използване

	• Дефиниране на Include-връзка между Use Cases • Дефиниране на Extend-Relationships между Use Cases • Дефиниране на Generalizations между Use Cases • Дефиниране на Generalizations между актьори 2. Управление на зависимостите между артефакти • Присвояване на атрибути на артефактите (сложност, риск, приоритет и др.) • Реализиране и проверка за проследяемост (Traceability) • Управление на променящите се изисквания
	Резултат: • Модела на начините на използване (актуализиран)

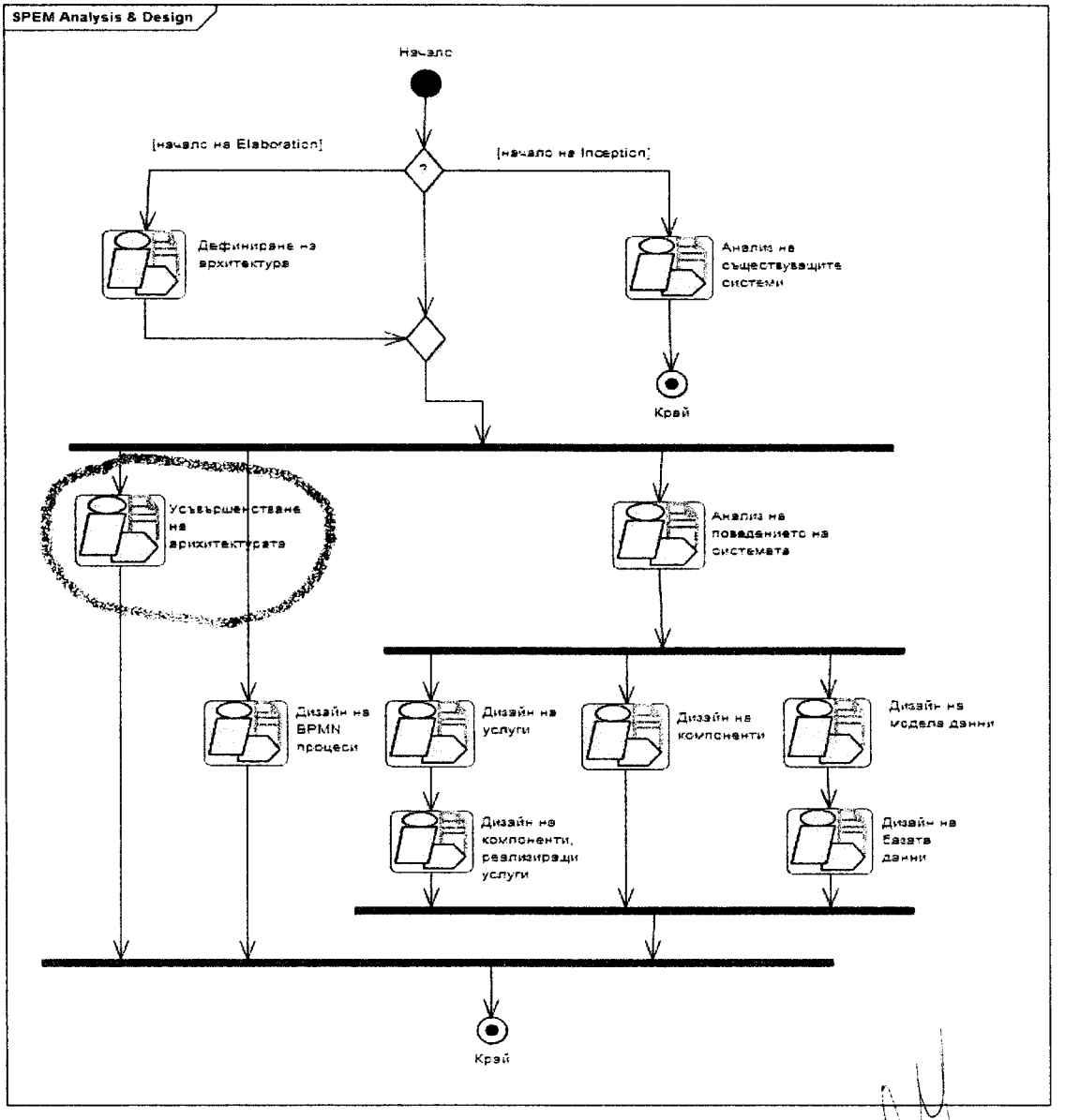
Дейност: Управление на обхвата на системата	
	Отговорна роля: Системен анализатор
	Вход: • Модел на начините на използване
	Стъпки: 1. Приоритизиране на начините на използване • Дефиниране на подмножество от сценарии, които представляват важна част от функционалността на системата. 2. Дефиниране на архитектурните решения
	Резултат: • Модел на начините на използване • Архитектурни решения

Дейност: Финализиране на дефиницията на системата	
	Отговорна роля: Бизнес анализатор Дизайнер на потребителски интерфейс
	Вход: • Модел на начините на използване
	Стъпки: 1. Детайлизиране на софтуерните изисквания

	2. Детайлизиране на Use Case 3. Моделиране на потребителския интерфейс 4. Създаване на прототип на потребителския интерфейс
	Резултат: • Модел на начините на използване (подобрен) • Модел на потребителския интерфейс (storyboard) • Прототип на потребителския интерфейс

13.2.2 Дейности от дисциплина „Анализ и дизайн"

Основните дейности от дисциплина „Анализ и дизайн", които са застъпени във фаза „Реализация", са обобщени на следващата диаграма:



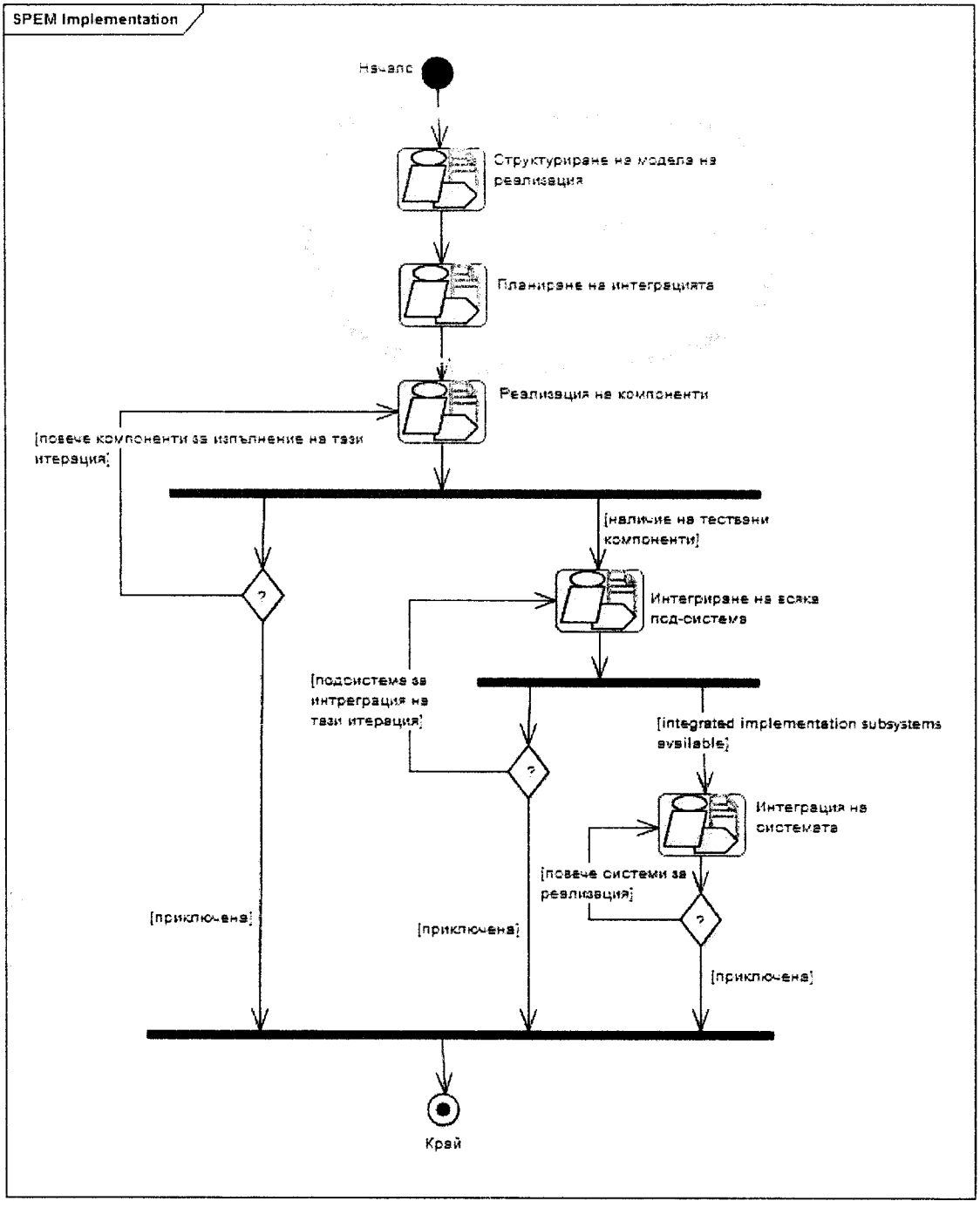
фигура 29

Описание на дейностите:

Дейност: Усъвършенстване на архитектурата	
	Отговорна роля: Софтуерен архитект
	Вход: • Обратна връзка от дейностите от дисциплина „Реализация” • Актуализиран Модел на начините на използване (Use Case Model) вследствие на постъпило изискване за промяна (change request) – резултат от изпълнение на следните дейности в дисциплина „Изисквания”/фаза „Реализация”: • Управление на променящите се изисквания • Управление на обхвата на системата • Финализиране на дефиницията на системата
	Стъпки: 1. Анализ на модела на начините на използване 2. Промяна на модела на BPMN2 процесите 3. Промяна на модела на услугите 4. Промяна на компонентния модел 5. Промяна на модела данни и базите данни
	Резултат: • Модел на BPMN2 процесите (усъвършенстван) • Модел на услугите (усъвършенстван) • Компонентен модел (усъвършенстван) • Модел на базите данни (усъвършенстван)

13.2.3 Дейности от дисциплина „Реализация”

Основните дейности от дисциплина „Реализация”, които са застъпени във фаза „Реализация”, са обобщени на следващата диаграма:



фигура 30

Описание на дейностите:

Дейност: Структуриране на модела на реализация	
	Отговорна роля: Софтуерен архитект
	Вход: • BPMN2 Модел за системата • Компонентен модел за системата

	Стъпки: 1. Създаване на първоначалната структура на модела на реализация 2. Дефиниране на подсистемите 3. Дефиниране на пакетите 4. Дефиниране на зависимостите между отделните пакети
	Резултат: • Модели на реализацията, съдържащи описание на подсистеми и пакети: • Модел на реализацията (implementation model)

Дейност: Планиране на интеграцията	
	Отговорна роля: Интегратор
	Вход: • Модел на реализациите на начините на използване (Use Case Realizations) • Модели на реализацията, съдържащи описание на подсистеми и пакети
	Стъпки: 1. Допълване на описанието на подсистемите 2. Дефиниране на "Build Sets" 3. Дефиниране на последователността на изпълнение на build
	Резултат: • План за интеграция (част от проектния план)

Дейност: Реализация на компоненти	
	Отговорна роля: Програмист
	Вход: • Модели на реализацията, съдържащи описание на подсистеми и пакети: • Модел на реализацията (implementation model) • Тестови модели за системата • Изискване за промяна (change request)

	Стъпки: 1. Реализация на компоненти • Реализация на методи • Реализиране на stateful компоненти • Реализация на асоциации • Реализация на атрибути • Обратна връзка към Дизайн • Усъвършенстване на кода 2. Реализация на тестови компоненти и подсистеми 3. Изпълнение на unit тестове 4. Премахване на дефект • Стабилизиране на дефект • Локализиране на проблема • Отстраняване на проблема
	Резултат: • Модел на реализацията (пълен) • Разработени подсистеми и компоненти на системата

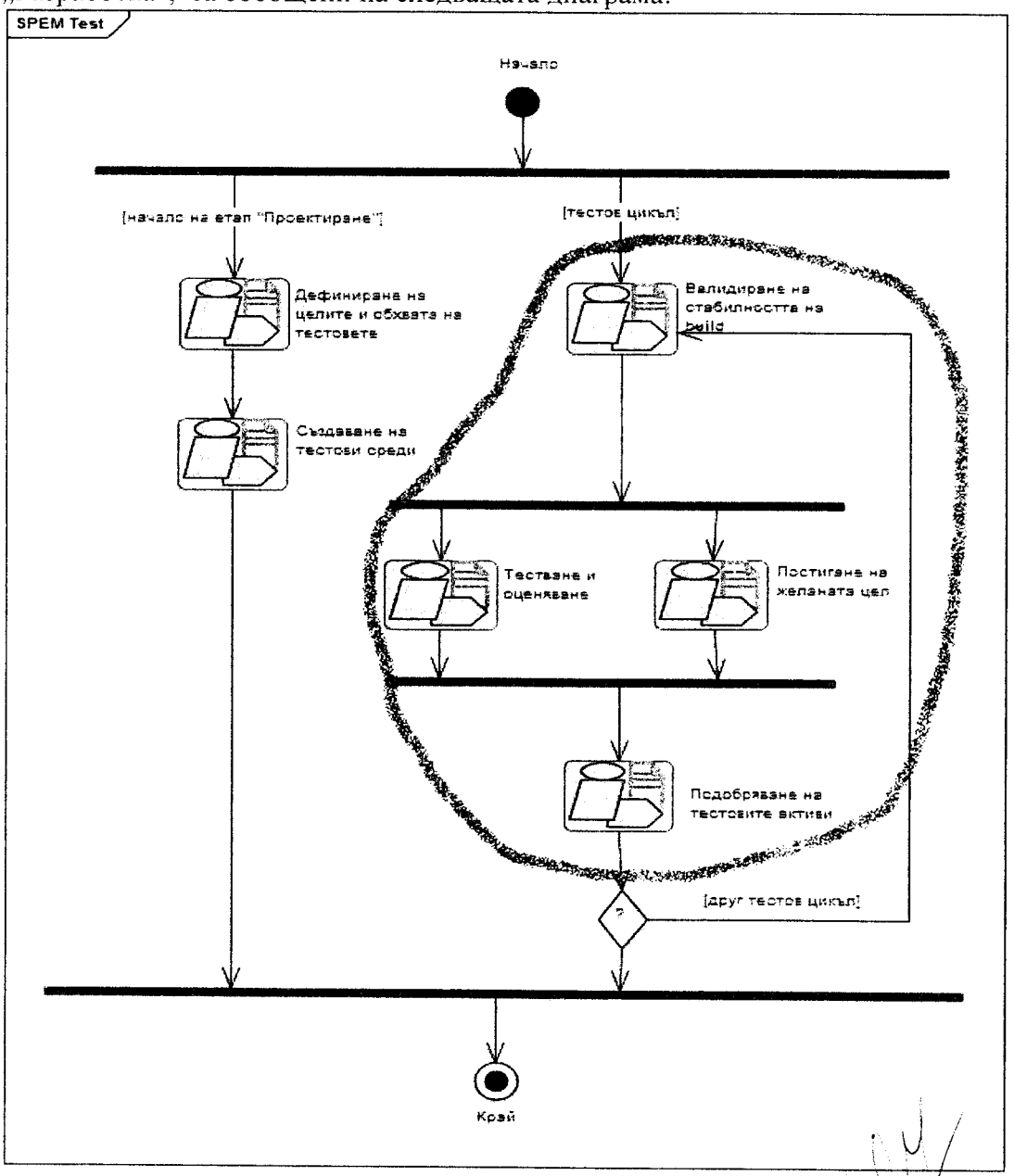
Дейност: Интегриране на всяка подсистема	
	Отговорна роля: Интегратор
	Вход: • Модел на реализацията • Разработени под-системи и компоненти на системата
	Стъпки: 1. Интегриране на компонентите 2. След всяка стъпка се създава билд и се провеждат интеграционни тестове. Когато финалната на стъпка тестовете преминат успешно то: 3. Доставя се подсистемата
	Резултат: • Функциониращи версии за софтуерния пакет

Дейност: Интеграция на системата

	Отговорна роля:
	Вход: • Функционираща версия на софтуерния пакет
	Стъпки: 1. Верифициране на реализираните подсистеми 2. Реализация на подсистеми с генеричен интерфейс 3. Интеграция на подсистемите
	Резултат: • Функциониращи версии за софтуерни пакети

13.2.4 Дейности от дисциплина „Тестване"

Основните дейности от дисциплина „Тестване", които са застъпени във фаза „Разработка", са обобщени на следващата диаграма:



фигура 31

Описание на дейностите:

Дейност: Валидиране на стабилността на build	
	Отговорна роля: Тест мениджър, Тест анализатор, Тестер
	Вход: • Функциониращи версии за софтуерни пакети
	Стъпки: 1. Дефиниране на детайли на теста 2. Дефиниране на очакваните резултати 3. Реализация на теста (тестов компонент: JBoss TestNG или друг) 4. Изпълняване на тестовете за всеки един софтуерен пакет
	Резултат: • Изпълнени тестове

Дейност: Тестване и оценяване	
	Отговорна роля: Тест анализатор
	Вход: • Резултати от изпълнени тестове
	Стъпки: 1. Анализ на резултатите от теста • Анализ на резултатите от проведените тестове • Идентифициране на процедурните грешки при изпълнение на тестовете • Локализиране и изолиране на грешки • Диагностициране на симптомите за грешки • Идентифициране на възможните решения на проблемите • Документиране на резултатите 2. Оценка за степента на стабилност на версиите • Анализ на проблемите по вид, честота и др. • Създаване на Change Requests в съответния проект в JIRA • Оценка за качеството на версията основана на метрики от

	тестовите резултати • Обсъждане на резултатите с разработчиците
	Резултат: • Изготвена оценка за степента на стабилност на версиите • Заявка за промяна (change request)

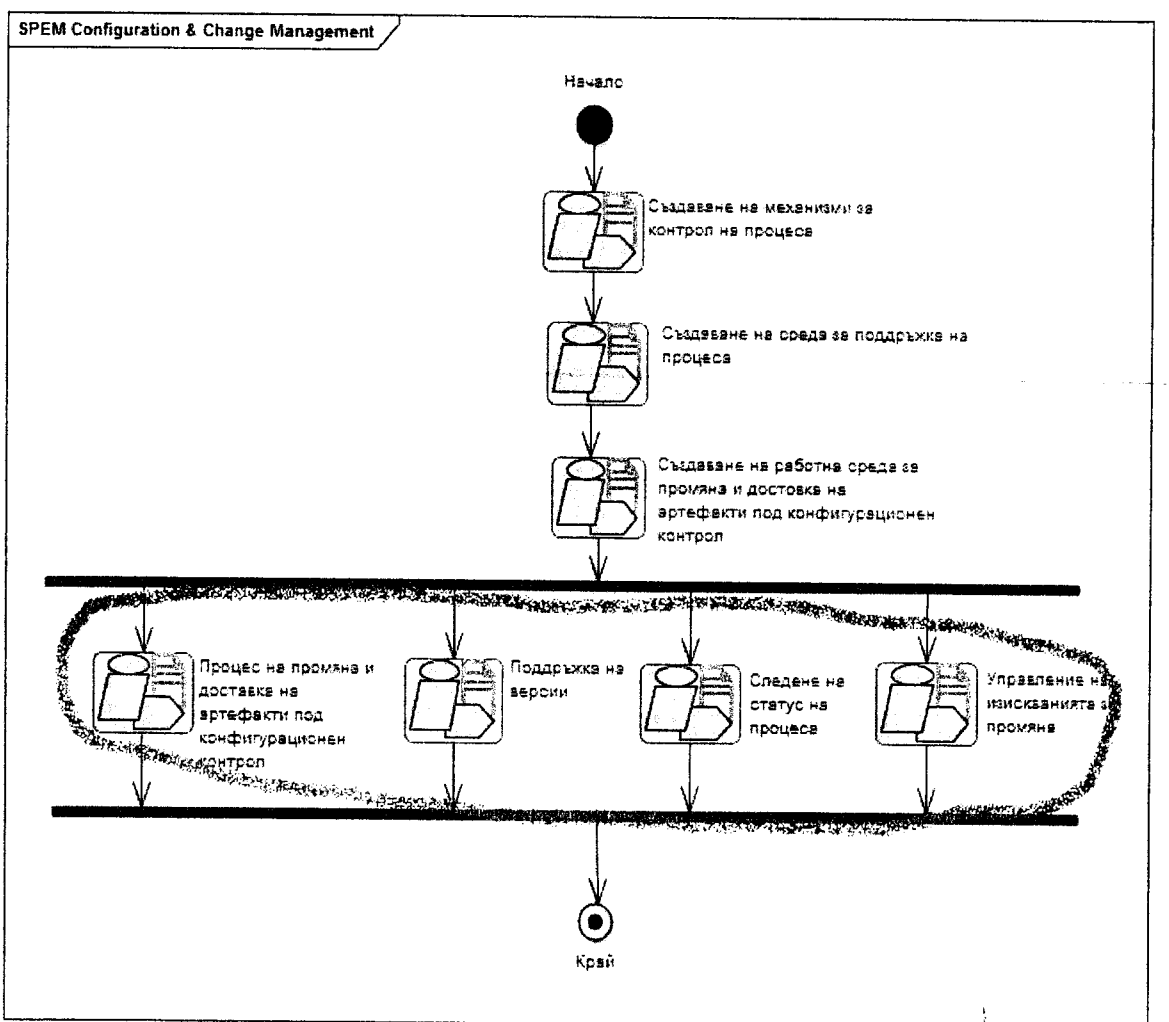
Дейност: Постигане на желаната цел	
	Отговорна роля:
	Вход: • Заявка за промяна (change request)
	Стъпки: 1. Оценка и подобряване на подхода за тестване • Следене на статуса на реализация • Следене на мерките за качество и ефективност на кода • Изготвяне на оценка • Планиране и осъществяване на мерки за подобрение на процеса 2. Оценка за качество • Извършване на статичен анализ на кода • Следене на мерките за качество • Анализ на заявките за промяна (change requests) • Идентифициране на причините за основните проблеми с качеството на кода • Дефиниране на мерки за решение на проблемите • Изпълнение на набелязаните мерки
	Резултат: • Регистрирани дефекти в съответния проект в JIRA • Изготвена оценка за качеството на предоставения код (Java)

Дейност: Подобряване на тестовите активи	
	Отговорна роля: Тест анализатор
	Вход: • Изготвена оценка за степента на стабилност на версиите

	• Изготвена оценка за качеството на предоставения код (Java)
	Стъпки: 3. Анализ на резултатите от проведените тестове 4. Идентифициране на слабите места и пропуски в процеса 5. Дефиниране на нови идеи/подходи за тестване 6. Адаптиране на тестовите сценарии към новите идеи и подходи
	Резултат: • Тестови сценарии за системата (адаптирани)

13.2.5 Дейности от дисциплина „Управление на конфигурацията и промените"

Основните дейности от дисциплина „Управление на конфигурацията и промените", които са застъпени във фаза „Разработка", са обобщени на следващата диаграма:



фигура 32

Описание на дейности:

Дейност: Процес на промяна и доставка на артефакти под конфигурационен контрол	
	Отговорна роля:
	Вход: • Задача (task)от СМ системата за реализиране
	Стъпки: Извършване на промени • Chek out на актуалната версия (създава локално копие на работната среда) • Извършване на промените Доставяне на промени • Тестване (unit) на направените промени • Update на извършените промени • Актуализиране на статуса на задачата в JIRA (in progress, completed, closed) Актуализиране на работната среда • Синхронизиране с актуалната версия от хранилището
	Резултат: Изпълнена задача (task)от СМ системата

Дейност: Поддръжка на версии	
	Отговорна роля: СМ мениджър Интегратор
	Вход: Неразрешени проблеми в JIRA
	Стъпки: Създаване на baseline (маркиране на група от артефакти, които са логически свързани и определят обхват на текуща версия) Създаване на deployment unit • Компилиране • Изпълнение на build скриптове

	Дефиниране на обхвата на нова версия • Дефиниране на списък с проблеми за отстраняване • Дефиниране на списък с изисквания за нова функционалност • Създаване на нова версия в CVS хранилището
	Резултат: Създаден baselinebaseline на текущата версия Дефиниран обхват на нова версия

Дейност: Следене на статус на процеса	
	Отговорна роля: CM мениджър
	Вход:
	Стъпки: Мониторинг на статуса • Генериране на справки за предложени промени и за статуса на тяхната реализация. • Справки за броя и в ида дефекти. • Справки за тенденциите Извършване на функционален конфигурационен одит (Functional Configuration Audit) Показва доколко един baseline покрива планираните изисквания. • Изготвяне на справка за всяко изискване, резултатите от тестването му (pass/fail). • Проверка за това дали всичко изискване е минало през процедура за тестване и дали всички тестове са били успешни. • Идентифициране на изискванията, които не са преминали през тестове или тестовете са били неуспешни • Генериране на списък с всички заявки за промяна. • Проверка за това дали всички заявки за промяна са затворени • Генериране на списък със заявките за промяна, които не са били реализирани. Извършване на физически конфигурационен одит (Physical Configuration

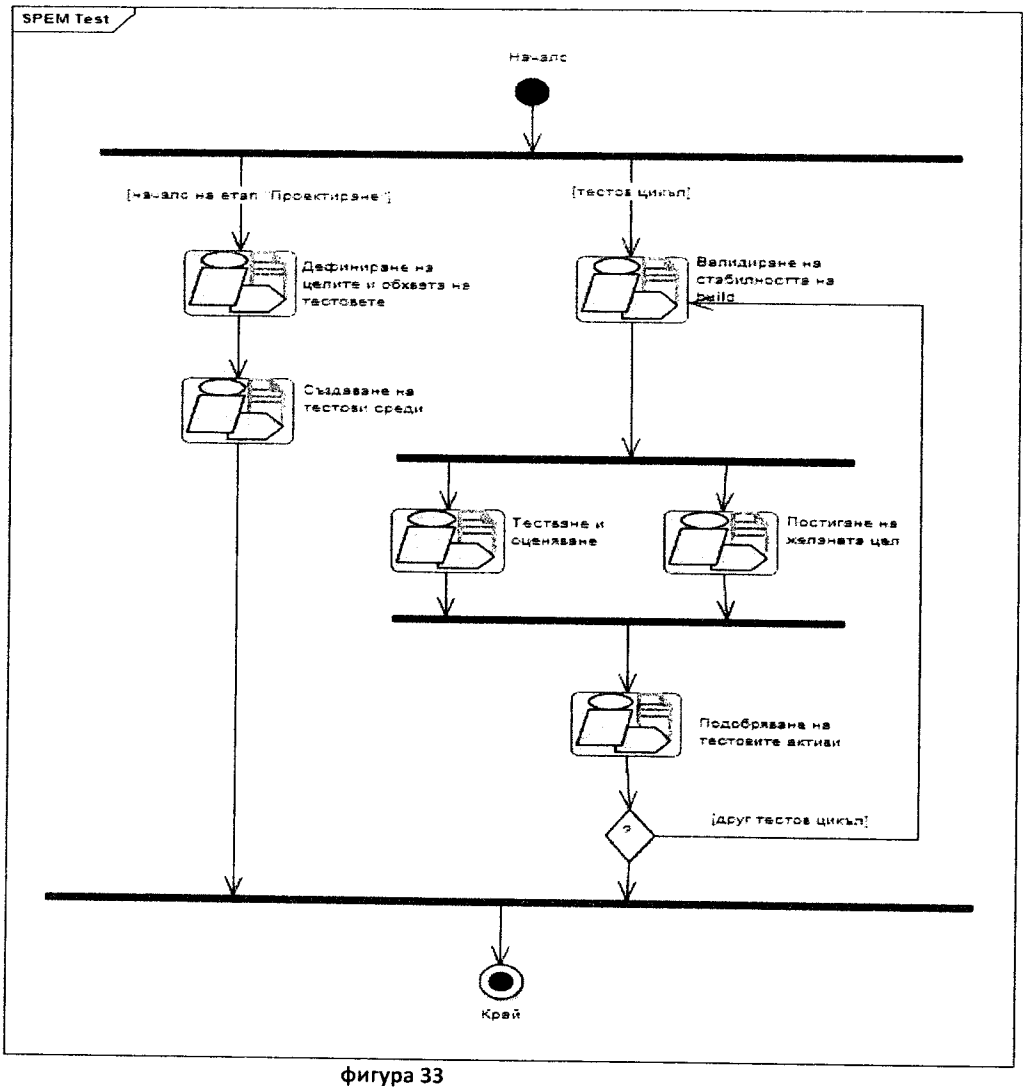
	Audit) Идентифицира компонентите, които трябва да бъдат инсталирани. • Идентифициране на версията, която трябва да бъде инсталирана • Проверка за наличие на всички зависимости (библиотеки, компоненти). • Генериране на списък с липсващите зависимости
	Резултат: Статус на изпълнение на проекта

Дейност: Управление на изискванията за промяна	
	Отговорна роля: Всяка роля СМ мениджър
	Вход:
	Стъпки: Подаване на заявка за промяна (Change Request) 1. Избор на проект в JIRA • Попълване на формата в JIRA • Подаване на заявката Преглед на подадените заявки за промяна • Класифициране на заявките (дефект, промяна) • Идентифициране на дублирани заявки • Отхвърляне на заявки • Промяна на статуса на заявката (new → not assigned →assigned)
	Резултат: Одобрени Заявки за промяна

13.3 Фаза Тестване

Според възприетия подход дейности за тестване има в 2 етапа: в етап „Разработка" и основно в етап „Предаване". Основната дисциплина е „Тестване". Целта на този етап е провеждането на тестове за приемане на системата. Тестването на предоставения продукт ще се извърши с цел да се провери функционалната коректност на

разработеното приложение и да се провери доколко то отговаря на очакванията на Възложителя.



фигура 33

Следва описание на дейностите:

Дейност: Дефиниране на целите и обхвата на тестовете	
	Отговорна роля: Тест дизайнер, Софтуерен архитект
	Вход: Избрана референтна архитектура за съответната система Компонентен модел Модел на начините на използване (Use Case Model) Модел на услугите BPMN2 модел на бизнес процесите

	Стъпки: Идентифициране на стратегия за използване на наличните ресурси (хардуер, хора) Дефиниране на оптимален обхват за тестовете Формално описание на процедурите за тестване (артефакти, дейности, отговорни роли) Дефиниране на механизъм за оценка на тестовите резултати и генериране на справки.
	Резултат: План за провеждане на тестове

Дейност: Създаване на тестови среди	
	Отговорна роля: Тест дизайнер, Софтуерен архитект
	Вход: План за провеждане на тестове Референтна архитектура на съответната система Компонентен модел Модел на потребителските сценарии Модел на услугите BPMN2 модел на бизнес процесите
	Стъпки: 1. Създаване на тестова среда за тестване на BPMN2 процеси 2. Създаване на тестова среда за тестване на интеграционни процеси
	Резултат: Тестова среда за тестване на BPMN2 процеси Тестова среда за Java приложения Създаване на тестова среда за тестване на интеграционни процеси

Дейност: Валидиране на стабилността на build	
	Отговорна роля: Тест мениджър, Тест анализатор, Тестер
	Вход:

	Функциониращи версии на кода

	Стъпки:
	1. Дефиниране на детайли на теста
	2. Дефиниране на очакваните резултати
	3. Реализация на теста (тестов компонент: JBoss TestNG или друг)
	
	В различните етапи от разработката на информационната система ще бъдат извършени различни видове тестове:
	
	Unit тестове – тестване, при което ще бъде създаден софтуерен код, който тества избран код на ниво функции и процедури. Ще бъде извършено от екипа по разработка.
	
	Функционални тестове –Този тип на тестване е базиран на техниката на черната кутия ("black box testing"), с която ще се проверяват шаблонните помощни софтуерни пакети през графичния потребителски интерфейс и ще се следят върнатите резултати. Целта на тези тестове е да се верифицира правилното приемане, получаване и преминаване на данните, както и правилното имплементиране. на бизнес логиката на системите.
	
	При провеждането на функционалните тестове ще бъдат съпоставяни действията и състоянията на системата с тези описани в спецификацията. Ще бъде проверен всеки един модул от системата. Ще бъде обърнато внимание на входните данни, навигацията, ролите и др.
	
	По време на функционалните тестове ще се провери дали двете реализирани системи на различни платформи имат:
	• Възможност за приемане на заявления за услуги електронни документи по всички канали – чрез уеб базирано приложение и на физически носител.
	• Изграден уеб базиран интерфейс за потребители, в който има възможност за:
	  • създаване и редактиране на електронни документи – заявления за електронни административни услуги;
	  • преглед на електронни документи – отговори от администрацията по административни услуги;
	  • визуализация и редактиране на електронните документи;
	  • достъп до статуса на стартирала услуга;
	  • достъп до друго съдържание, свързано с описания на процеса по предоставяне на административната услуга.

съгласно изискванията на нормативната уредба.

- Изграден уеб базиран интерфейс за служители на администрацията за:
  - обработка на постъпили заявления за електронни административни услуги;
  - съставяне на електронни документи, отговори по електронни административни услуги;
  - визуализация и редактиране на електронните документи;
  - административен модул за настройка, конфигуриране и наблюдение на системните параметри

Интеграционни тестове – ще бъдат проведени тестове за интеграция на модулите (при необходимост) с:

- бек офис информационни система;
- системи за управление на потребители за целите на автентикацията и оторизация на потребителите;
- Други вътрешни системи на администрацията за обмен на ЕД.

Тестване на производителност – ще бъдат създадени различни тестове, целта на които е да оценят работата на системата при различно нейно натоварване. При тези тестове се следят различни характеристики на системата като време за отговор, едновременно работещи потребители и др. В частност ще бъдат направени стрес тестове на системата, които ще бъдат извършени, за да се оцени как се държи тя при натоварване по голямо от предвиденото. Целта им ще бъде да се намери пределната точка на натоварване, след която системата не би могла да функционира коректно.

Тестване на сигурността и контрола на достъпа - Този тип тестване се фокусира върху сигурността в системата. Ще бъдат проверени дейностите, които системата предоставя на различните типове потребители, а именно граждани/организации, служителите от администрациите, администраторите на системата. При тестове ще бъде проверено дали правата получени от всяка от тези различни групи не се припокриват.

Тестове за приемане на системите – Тези тестове ще бъдат извършени от Възложителя с помощта на Изпълнителя, като предварително бъдат избрани от двете страни достатъчен набор от функционални тестове, с които да се демонстрира коректната работоспособност на системите.

	4. Изпълняване на тестовете за всяка една от версиите на шаблонен помощен софтуерен пакет
	Резултат: Изпълнени тестове

Дейност: Тестване и оценяване	
	Отговорна роля: Тест анализатор
	Вход: • Тестови сценарии • Резултати от изпълнени тестове
	Стъпки: 1. Анализ на резултатите от теста • Анализ на резултатите от проведените тестове • Идентифициране на процедурните грешки при изпълнение на тестовете • Локализиране и изолиране на грешки • Диагностициране на симптомите за грешки • Идентифициране на възможните решения на проблемите • Документиране на резултатите 2. Оценка за степента на стабилност на версиите • Анализ на проблемите по вид, честота и др. • Създаване на Change Requests в съответния проект в JIRA • Оценка за качеството на версията основана на метрики от тестовите резултати • Обсъждане на резултатите с разработчиците
	Резултат: • Изготвена оценка за степента на стабилност на версиите • Заявка за промяна (change request)

Дейност: Постигане на желаната цел	
	Отговорна роля:
	Вход: • Заявка за промяна (change request)
	Стъпки: 1. Оценка и подобряване на подхода за тестване • Следене на статуса на реализация

	• Следене на мерките за качество и ефективност на кода • Изготвяне на оценка • Планиране и осъществяване на мерки за подобрение на процеса 2. Оценка за качество • Извършване на статичен анализ на кода • Следене на мерките за качество • Анализ на заявките за промяна (change requests) • Идентифициране на причините за основните проблеми с качеството на кода • Дефиниране на мерки за решение на проблемите • Изпълнение на набелязаните мерки
	Резултат: • Регистрирани дефекти в проекта в JIRA • Изготвена оценка за качеството на предоставения код

Дейност: Подобряване на тестовите активи	
	Отговорна роля: Тест анализатор
	Вход: • Изготвена оценка за степента на стабилност на версиите • Изготвена оценка за качеството на предоставения код
	Стъпки: 1. Анализ на резултатите от проведените тестове 2. Идентифициране на слабите места и пропуски в процеса 3. Дефиниране на нови идеи/подходи за тестване 4. Адаптиране на тестовите сценарии към новите идеи и подходи
	Резултат: • Тестови сценарии (адаптирани)

14 ИЗПОЛЗВАНИ ТЕХНОЛОГИИ И ТЕХНИЧЕСКИ СРЕДСТВА

14.1 Технологии

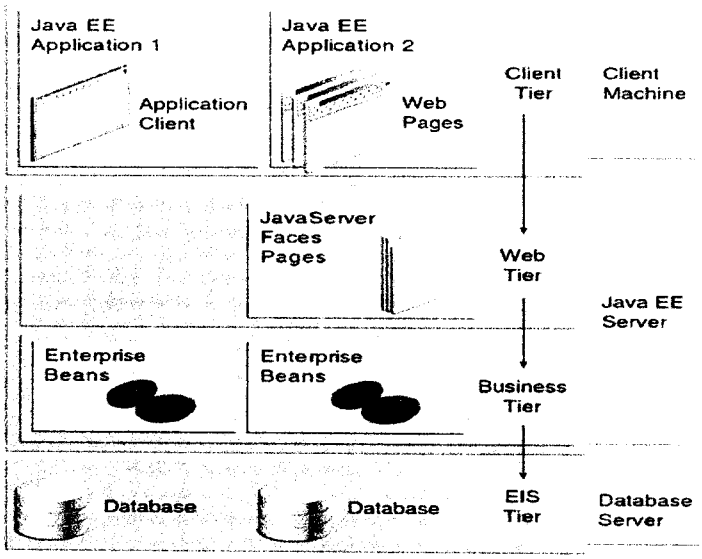
14.1.1 Java EE

Java Platform. Enterprise Edition или Java EE е широко използвана платформа за сървър програмиране на езика за програмиране Java. Платформата Java (Enterprise Edition) се различава от Java Platform Standard Edition (Java SE) в това, че добавя библиотеки и програмни интерфейси, които предоставят функционалност за създаване на отказо устойчива, разпределена, многослойна Java архитектура, която се основава главно на модулни компоненти и работи на сървър на приложения.

Програмни интерфейси, предоставяни от Java EE 6:

- Enterprise JavaBeans Technology;
- Java Servlet Technology;
- JavaServer Faces Technology;
- JavaServer Pages Technology;
- JavaServer Pages Standard Tag Library;
- Java Persistence API;
- Java Transaction API;
- Java API for RESTful Web Services;
- Managed Beans;
- Contexts and Dependency Injection for the Java EE Platform (JSR 299);
- Dependency Injection for Java (JSR 330);
- Bean Validation;
- Java Message Service API;
- Java EE Connector Architecture;
- JavaMail API;
- Java Authorization Contract for Containers;
- Java Authentication Service Provider Interface for Containers.

Java EE предоставя следните контейнерни услуги (контейнер представлява интерфейс между компонент и системна, платформено-зависима функционалност, която поддържа функционирането на компонента):



фигура 34

Java EE server - предоставя EJB и уеб контейнери.

Enterprise JavaBeans (EJB) container - управлява изпълнението на EJB компонентите в Java EE приложенията.

Web container - управлява изпълнението на уеб страници, Servlets, и някои EJB компоненти за приложения на Java EE.

Application client container - управлява изпълнението на компонентите на клиентските приложения.

Applet container - управлява изпълнението на аплети.

14.1.2 SAML

Security Assertion Markup Language (SAML) е стандартизиран език на основата на XML и се използва за обмен на данни за автентикация и оторизация между защитени домейни, в частност между доставчик на електронни услуги и консуматор на такива услуги. В допълнение SAML е и набор от протоколи, обвързвания (binding) и профили (profiles). Едно от основните приложения на SAML е реализация на механизъм за еднократна автентикация (Single Sign-On (SSO)).

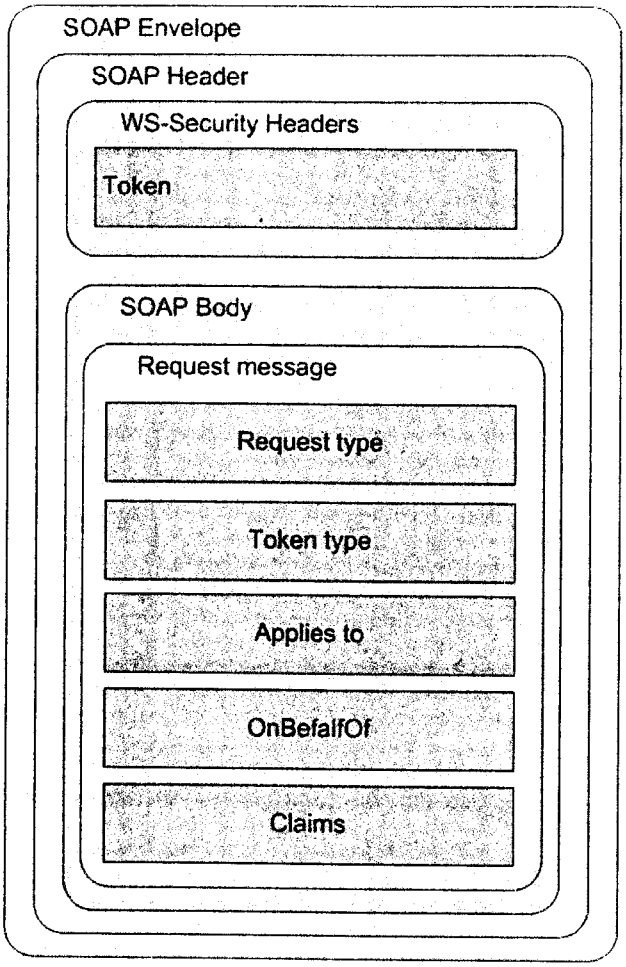
14.1.3 WS-Trust

WS-Trust е WS-* спецификация и стандарт, разработен от организацията OASIS, който представлява разширение на спецификацията WS-Security.

Използва се за издаване, подновяване и валидиране на токени, които от своя страна се използват за нуждите на автентикация между две системи, които обменят данни помежду си.

За издаване на токен се използва съобщението RequestSecurityToken Request (RSTR).

Структурата на RSTR съобщението е изобразена на следващата фигура:

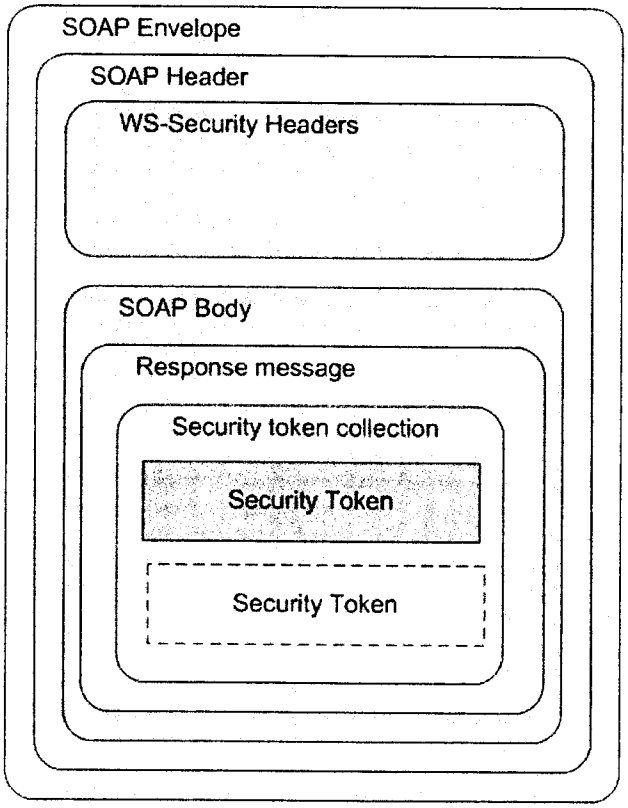


Фигура 35 WS-Trust RST съобщение

Част от заявката	*Описание*
Request type	`http://docs.oasis-open.org/ws-sx/ws-trust/200512/Issue`
Token type	Вид на токена, който се заявява: Може да бъде, например: *Вид на токен* — *Стойност на елемента* Username Token — http://docs.oasis-open.org/wss/2004/01/oasis-200401-wsswssecurity-secext-1.0.xsd/UsernameToken SAML V2.0 assertion — http://docs.oasis-

	open.org/wss/oasis-wss-saml-token-profile-1.1#SAMLV2.0
Applies to	Определя за кои системи този токен може да бъде използван за автентикация. Това е адресът на доставчика на услуги
OnBefalfOf	Определя, че инициаторът на WS-Trust заявката заявява SAML токен за друг субект.
Claims	Задава допълнителни характеристики за субекта, които трябва да присъстват в издавания токен.

Резултатът от заявката има следния вид:



Фигура 36 WS-Trust оговор на RST

Резултатът съдържа един или повече токени от заявения вид , всеки един тях в елемент: `/wst:RequestSecurityTokenResponse/wst:RequestedSecurityToken/wsse:BinarySecurityToken`

14.1.4 UDDI

UDDI (Universal, Description, Discovery and Integration) е платформено независим регистър на услуги, предлагани от информационни системи. Услугите са категоризирани според поддържани таксономии. UDDI е отворена инициатива, позволяваща на различните видове бизнес да публикуват списъци с услуги, да се намират лесно и да дефинират как дадена услуга или софтуерни приложения взаимодействат с интернет. UDDI е един от основните стандарти за уеб услуги. Разработен е да бъде използван от SOAP съобщения и да предоставя достъп до WSDL (Web Services Description Language) документи, описващи протоколни свързвания и формати за съобщения, изисквани за взаимодействие с уеб услуги.

В инфраструктурата на БеУ има инсталиран регистър UDDI регистър, в който са ще се дефинират всички публично достъпни уеб услуги, предлагани от администрациите..

Комбинацията от ESB и UDDI позволява реализирането на т. нар. виртуализиране на услуги. В регистъра се пази връзката между система, УРИ на АИС и точката за достъп. По този начин при инсталиране на нова АИС и/или преместване на някоя система на нов адрес, се променя само точката за достъп (access point) в UDDI регистъра без да е необходимо пренаписване на части от приложенията.

14.2 Технически средства

14.2.1 Eclipse IDE

Eclipse е многоезична среда за разработка на софтуер , включваща интегрирана среда за разработка (IDE). Използва се основно за разработка на приложения на Java, но приложения могат да бъдат разработвани и на други езици, като: Ада, C, C + + , COBOL , Perl , PHP , Python , Ruby (включително Ruby), Scala , Clojure.

Eclipse е свободен софтуер с отворен код и се разпространява съгласно условията на Eclipse Public License.

Eclipse използва т.нар. плъгини (plugin), които предоставят цялата функционалност в рамката на единна работа среда. Eclipse реализира OSGi спецификацията въз основа на проекта Equinox.

Eclipse предоставя следните инструменти за разработка и стандарти:

- Standard Widget Toolkit (SWT)
- JFace
- SWT
- UML
- OCl
- BPMN
- IMM
- SBVR
- XSD

За реализация на компонентите на Java ще се използва набор от плъгини/добавки за Eclipse, по-важните от който са:

- JBoss Tools. Този набор от инструменти поддържат технологии, като: EJB3, Hibernate, JBoss AS, Drools, jBPM, JSF, (X)HTML, Seam, Smooks, JBoss ESB, JBoss Portal и други;
- M2E. Осигурява поддръжка на maven инструменти за конструиране в Eclipse;
- Subclipse или Subversive. Предоставя интеграция със SVN системи за контрол на версии на изходния код;
- Activiti или Camunda Modeler за визуална редакция на BPMN процеси;
- IBM ODM business rules designer;
- IBM ODM business events designer;
- IBM WID за разработка на ESB mediation модули и примитиви.

14.2.2 Apache CXF STS

Apache CXF STS е реализация на Security Token Service (STS). STS представлява набор от средства за издаване на токени за автентикация в случаите, когато е необходим надежден обмен на данни между системи. STS предоставя стандартен WS-Trust интерфейс.

14.2.3 JBoss SEAM

JBoss SEAM е платформа, която реализира модел-изглед –контролер архитектурен шаблон (MVC) и е разширение на Java спецификацията JSF. JBoss SEAM дефинира общ компонентен модел за едно приложение и предоставя иновационна рамка на сигурност, основана на правила върху JAAS. Предоставя се поддръжка на PDF, изходяща поща, графики и wikitext. Компонентите на JBoss SEAM могат да бъдат извикани синхронно като Web Service, асинхронно от клиентски JavaScript или Google Web Toolkit, или разбира се директно от JSF.

14.2.4 JBoss Application Server

JBoss AS е приложен сървър с отворен код, основаващ се на стандарта J2EE. JBoss AS осигурява среда за работата на сървлети и EJB (Enterprise Java Beans) базирани приложения, предоставяйки както уеб, така и EJB контейнер.

Основни характеристики на продукта са:

- клъстериране, разпределение на натоварването (load balancing),
- разпределен caching
- разпределен deployment
- поддръжка на аспектно-ориентирано програмиране (AOP)
- JSP/Servlet 2.1/2.5
- JavaServer Faces 1.2
- Enterprise Java Beans версия 3 и версия 2.1
- JNDI (Java Naming and Directory Interface)

- JPA
- JDBC
- JTA (Java Transaction API)
- поддръжка на уеб сървиси (JAX-WS)
- SAAJ (SOAP with Attachments API for Java)
- Интеграция на JMS (Java Message Service) integration
- JavaMail
- RMI-IIOP (JacORB, alias Java and CORBA)
- JAAS (Java Authentication and Authorization Service)
- JCA (Java Connector Architecture)-integration
- Интеграция на JACC (Java Authorization Contract for Containers)-integration
- Java Management Extensions (JMX)

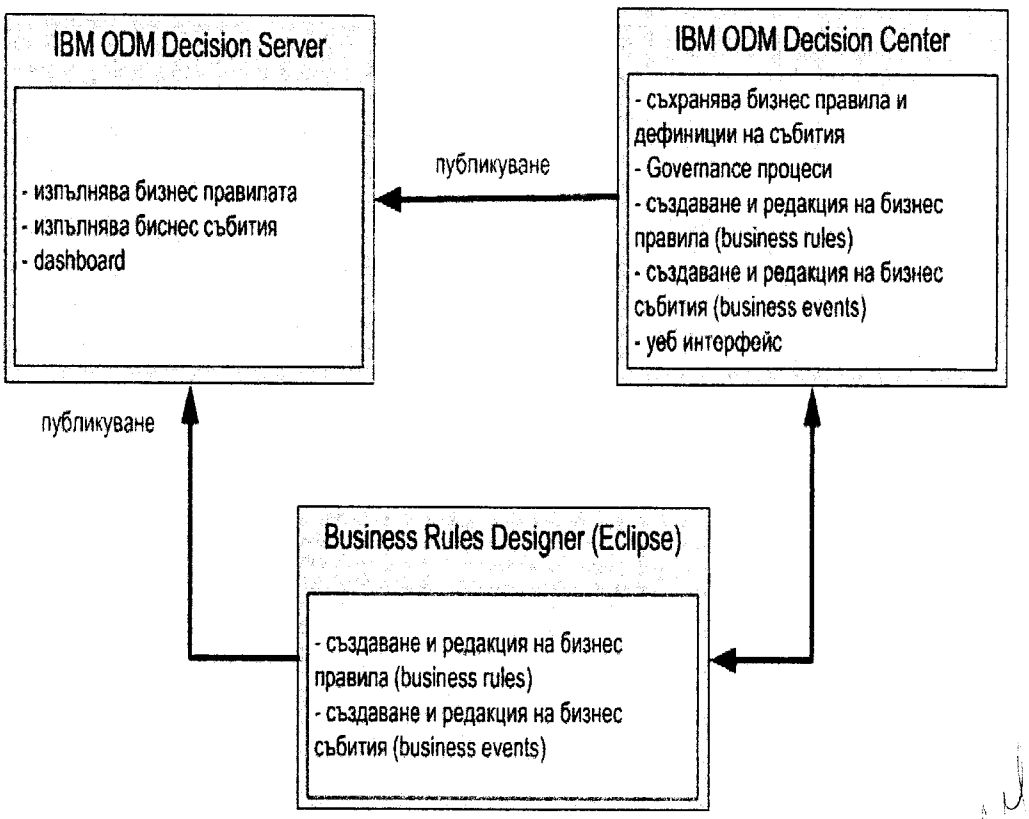
14.2.5 Activiti BPM

Платформа с отворен код за дизайн, публикуване, изпълнение и поддръжка на бизнес процеси, описани с езика BPMN.

14.2.6 IBM Operational Decision Management

IBM Operational Decision Management (IBM ODM) е платформа за дизайн, публикуване, изпълнение и поддръжка на бизнес правила и бизнес събития.

IBM ODM се състои от следните основни компоненти:



Фигура 37 Системи, от които е изграден продуктът IBM ODM

Платформата ще се използва при реализацията на два основни компонента в инфраструктурата на БеУ:

- Компонента за електронна оторизация (еОтор) (т. 8)
- Системата за обработка и генериране на бизнес събития (т. 10)

IBM ODM ще е вграден компонент на посочените по-горе решения, като в тази връзка лицензите, които ще се ползват са от тип OEM. Този тип лицензиране е само за разработка на нови решения и в конкретният случай могат да се използват само за нуждите на изброените по-горе решения.

Фирма Бул Ес Ай ще осигури необходимите лицензи за продукта за нуждите на разработваните и предлагани решения.

Дата 25.08.2014 г.

Управител: