

8.Zmeny verejného API medzi verziami 10.0.37 a 10.1.39

- 8.1. Odlenenie balíka `sk.ipesoft.d2000.async`
- 8.2. Zabezpečené a reverzné pripojenie JAPI
- 8.3. Zdieané zdroje systému D2000
- 8.4. Použitie konkrétnych `Table` tried namiesto `AnonymousTable`
- 8.5. Podpora odahenej verzie štruktúrovanej UNIVAL hodnoty
- 8.6. Rozšírenie itania dát z archívu o `ArchiveObjectDescription`
 - 8.6.1. Rozšírenie rozhrania `ArchiveDataListener`
 - 8.6.2. Rozšírenie rozhrania `ArchiveDataProvider`
- 8.7. Výpočet archívnej štatistickej funkcie
- 8.8. Rozšírenie rozhrania `AuditDataListener`
- 8.9. Úprava logiky operácie `close` v rozhraniach `D2Object`, `D2Cono`, `D2ProcessList`, `D2Session`
- 8.10. Rozšírenie rozhrania `D2SessionEventsListener`
- 8.11. Volanie RPC implementujúcej ESL rozhranie
- 8.12. Test existencie RPC a SBA volania
- 9.13. Informácie o prihlásenom používateľovi a zmena hesla používateľa
- 9.14. Zmena správania pri neúspešnom vytvorení `D2Session`
- 9.15. Volanie `D2Session.getIndirectInfo`
- 9.16. Volanie `D2Session.getObjectInfo` vráti aktuálnu hodnotu objektu
- 9.17. Volanie `D2Session.openObject` vráti informácie o objekte
- 9.18. Zmena rozhrania `RPCReturnContext`
- 9.19. Zmena rozhrania `SBAReturnContext`
- 9.20. Zastaraná `D2SessionUtils.getObjectHOBJByName`

Verejné API sa medzi verziami 10.0.37 (`d2japi-37.jar`) a 10.1.39 (`d2japi-10.1.39.jar`) rozšírilo a v niektorých prípadoch aj zmenilo natoko, že sú pri prechode na novšiu verziu potrebné ručné úpravy existujúceho kódu. (Pozn. verzia 10.1.38 bola vydaná len pre vnútorné potreby a nie je jej v tejto migranej príručke venovaná pozornosť.) Nasledujúcich kapitolách je uvedený zoznam zmien, nových funkcií a návod, ako správne upravi aplikovaný kód pri migrácii aplikácie z D2000 verzie 10.0.37 na 10.1.39.

8.1. Odlenenie balíka `sk.ipesoft.d2000.async`

Pre lepšie logické usporiadanie tried bol vytvorený balík `sk.ipesoft.d2000.async` ktorý obsahuje niektoré rozhrania, ktoré sa predtým nachádzali v balíku `sk.ipesoft.d2000.d2japi`. Prehľad zmien dokumentuje nasledujúca tabuľka:

trieda/rozhranie	vo verzii 10.0.37	vo verzii 10.1.39
<code>FutureEvent</code>	<code>sk.ipesoft.d2000.d2japi</code>	<code>sk.ipesoft.d2000.async</code>
<code>FutureEventHandler</code>	<code>sk.ipesoft.d2000.d2japi</code>	<code>sk.ipesoft.d2000.async</code>
<code>AdjustableFuture</code>	<code>sk.ipesoft.d2000.d2japi.common</code>	<code>sk.ipesoft.d2000.async.base</code>



Pri migrácii je potrebné upravi všetky príkazy import, ktoré odkazujú na spomínané triedy/rozhrania a všetky absolútne odkazy.

8.2. Zabezpečené a reverzné pripojenie JAPI

Trieda `D2Japi` predstavuje vstupný bod do používania knižnice JAPI. Používa sa na pripojenie sa k bežiacemu D2000 kernelu. Pripojenie sprostredkuje proces `D2Connector.exe`. Vo verzii 10.0.37 bolo možné pripojenie sa iba jednou metódou `createConnector` (`String`, `D2ConnectorEventsListener`). Pripojenie inicioval klient JAPI a pripájal sa na používajúci socket procesu.

Verzia 10.1.39 pridala metódu `startListeningForConnection(String, int, ServerSocketEventsListener)`. Metóda otvorí používajúci socket na strane JAPI klienta, ktorý aká na pripojenie iniciované procesom `D2Connector.exe`.

Od verzie 10.1.39 je možné zabezpečiť komunikáciu medzi JAPI klientom a `D2Connector.exe` protokolom TLS verzie 1.2. Slúžia na to preaznenia metód `createConnector` a `startListeningForConnection` s parametrom `String certificatePath`, ktorý uvádza cestu k súboru s certifikátom, ktorý je použitý na zabezpečenie spojenia.

8.3. Zdieané zdroje systému D2000

Zdieané zdroje systému D2000 sú

- stavové texty,
- systémové texty (pomenovania konštánt),
- transformovaná paleta,
- slovník,

- definície štruktúr.

Od verzie 10.1.39 je možné získať ich aktuálnu konfiguráciu a notifikácie o zmenách hodnôt. Služi na to preaznenie metódy `D2Connector.createSession` s parametrom `SharedResourcesListener` `sharedResourcesListener`.

8.4. Použitie konkrétnych Table tried namiesto AnonymousTable

Štruktúrovaná UNIVAL hodnota je v knižnici JAPI reprezentovaná objektom typu `UnivalRecord`. Atribúty UNIVAL-u sú v tomto prípade málo zaujímavé, hodnota je ale zabalená objektom triedy `Table`, ktorý je sprístupnený volaním `getValue`. Trieda `sk.ipesoft.d2000.datatable.Table` je abstraktná a pre jednoduchú tvorbu inštancií slúži odvodená trieda `AnonymousTable`, ktorú v základnej konfigurácii používa aj knižnica JAPI pri dekódovaní. Nevýhodou tejto triedy je, že neponúka prístup k jednotlivým bunkám pomocou identifikátora stĺpca a neumožňuje ich statickú typovú kontrolu.

Od verzie 10.1.39 je možné volaním `D2Connector.registerTableClass` zaregistrovať vlastnú implementáciu triedy `Table` (odporúčané je dedičstvo z triedy `TableBase`). Konkrétna implementácia vždy reprezentuje jednu konkrétnu definíciu štruktúry s ktorou je spojená anotáciou `@StructureTableDefinition`. Po takejto registrácii bude JAPI dekódovať príslušné štruktúrované hodnoty ako inštancie registrovanej triedy namiesto inštancií `AnonymousTable`.



Pozor: Po takejto registrácii sa JAPI pri dekódovaní spolieha na to, že počet a typy stĺpcov registrovanej triedy zodpovedajú definícii štruktúry. Ak sa odlišujú, dekódovanie pravdepodobne skoní výnimkou. K odlišnosti môže dôjsť napríklad aj neskoršou zmenou konfigurácie definície štruktúry v nástroji CNF. V takom prípade je potrebné implementáciu príslušnej triedy upraviť a aplikáciu nanovo skompilovať.

8.5. Podpora odahenej verzie štruktúrovanej UNIVAL hodnoty

V súvislosti s vývojom projektu *D2000 Web Suite* bola knižnice podporená konverzia inštancie triedy `Table` na `List<TableRowBean>` a späť. `TableRowBean` je abstraktná trieda predstavujúca odahenú štruktúrovanú hodnotu s rozhraním zodpovedajúcim konvencii Java Beans. Pre dosiahnutie dobrej statickej typovej kontroly kódu bol rozšírený zoznam generických parametrov viacerých tried. Vo všetkých uvedených triedach ako aj v ich potomkoch je nový generický parameter definovaný ako `B` extends `TableRowBean`.

- `Table<T, R, B>`
- `TableRow<T, R, B>`
- `Column<I, S, T, R, B, C, E>`
- `TableCell<I, S, T, R, B, C, E>`



Pri migrácii označte všetky miesta s nesprávnym potom generických parametrov. Oprava spoíva v doplnení chýbajúceho generického parametra:

- Pri získaní odkazu na existujúci objekt bez známeho typu pridaním „wildcard“ parametra: `Column<?, ?, ?, ?, ?, ?, ?>`
`column = record.getValue().getColumn(1);`
- Pri vytváraní objektu špecifikovaním všeobecného typu `TableRowBean`: `UnivalRecord<AnonymousTable, AnonymousTableRow, TableRowBean> record = new UnivalRecordValue<>(...);`

8.6. Rozšírenie íťania dát z archívu o ArchiveObjectDescription

Íťanie dát z archívu bolo rozšírené o konfiguráciu príslušného archívneho bodu. Konfigurácia je zapuzdrená rozhraním `ArchiveObjectDescription`. Z objektu je možné preíť informácie o názve objektu, jeho popise, technických jednotkách, type hodnoty, periodicite a iných informáciách potrebných pre zobrazenie hodnoty pravidlami systému D2000.

8.6.1. Rozšírenie rozhrania ArchiveDataListener

Pribudla metóda `void onArchiveObjectDescription(ArchiveObjectDescription)`, ktorou implementátor získava konfiguráciu. Ak nenastala chyba pri prístupe do archívu, metóda je zavolaná vždy ako prvá. Samotné dáta z archívu sú volaním metódy `onArchiveData` sprístupnené až následne.



Pri migrácii je potrebné rozšíriť všetky implementácie rozhrania `ArchiveDataListener` o spomínanú metódu. Implementácia metódy môže byť prázdna.

8.6.2. Rozšírenie rozhrania ArchiveDataProvider

Rozhranie `ArchiveDataProvider` umožňuje alternatívny prístup k dátam z archívu. Bolo rozšírené o metódu `getDescription()`, ktorá sprístupňuje konfiguráciu príslušného archívneho bodu. Táto zmena je spätne kompatibilná.

Okrem toho boli obidve preaznenia metódy `waitForData()` doplnené o kontrolované výnimky `InterruptedException`, `D2JapiException`, ktoré umožňujú zachytiť výnimkové stavy, ktoré pri získaní dát mohli nastať. Je to zmena správania oproti predchozej verzii, kde boli tieto výnimkové stavy v tichosti zamľané.

i Pri migrácii je potrebné blok kódu, ktorý tieto dáta spracúva obali do try... catch... alebo deklarovať kontrolované výnimky v metóde, ktorá k dátam pristupuje.

8.7. Výpočet archívnej štatistickej funkcie

Od verzie 10.1.39 je podporené volanie archívnych štatistických funkcií z JAPI. Ide o ekvivalent ESL akcie CALCSTATFUNCARR. Služí na to volanie `DSession.getArchiveValuesThroughStatisticalFunction`.

8.8. Rozšírenie rozhrania `AuditDataListener`

Rozhranie `AuditDataListener` bolo rozšírené o metódu `void onError(String comment)`, ktorá je zavolaná, ak pri prístupe k dátam došlo k chybe.

i Pri migrácii je potrebné doplniť všetky implementácie o danú metódu. Implementácia metódy môže byť prázdna, ale jej zavolaním sa signalizuje ukonenie zasielania samotných dát.

8.9. Úprava logiky operácie `close` v rozhraniach `D2Object`, `D2Cono`, `D2ProcessList`, `D2Session`

Tieto rozhrania sú si navzájom podobné v tom, že reprezentujú určitý zdroj platformy D2000, ktorý v ňom mení svoj stav a registrovaným klientom o tom v reálnom ase posiela notifikácie. Posiela ich tak dlho, kým klient nepožiadá o ukonenie odberu notifikácií, alebo kým daný zdroj nezanikne. Za distribúciu notifikácií ako aj udržiavanie zoznamu registrovaných odberateľov je zodpovedný Kernel. (V prípade CONO je zodpovedný samotný proces produkujúci správy). V knižnici JAPI slúži na zrušenie registrácie v týchto rozhraniach metóda `close()`.

Vo verzii 10.0.37 sa kanál považoval za uzatvorený zavolaním metódy `close()` okamžite. To však nezodpovedá stavu v systéme D2000, pretože zavolaním metódy sa len odosiela správa so žiadosťou o zrušenie registrácie. V ase medzi zavolaním `close()` a spracovaním správy kernelom ešte môžu prichádzať notifikácie o zmene hodnoty.

Z toho dôvodu je signatúra metódy vo verzii 10.1.39 `FutureEvent<Void> close()`. Volaním operácie sa kanál neuzatvorí automaticky, ale až vtedy, keď je doručená odpoveď potvrdzujúca uzatvorenie. Dovtedy ešte stále môžu prichádzať notifikácie. Doručenie potvrdenia je signalizované tým, že návratový `Future` objekt nadobudne hodnotu.

i Pri migrácii je nutné sa uistiť, že po zavolaní `close()` sa následné príkazy nespoliehajú, že bol kanál okamžite uzatvorený. Pokiaľ je v kóde takéto miesto, je potrebné pred ním spraviť nasledovné:

```
FutureEvent<Void> futureCloseResult = openedObject.close();

... // tento kód sa na uzatvorenie ešte nespolieha

futureResult.get(); // akanie na potvrdenie uzatvorenia

... // nasleduje miesto, na ktorom už musí platiť, že bol kanál
skutone uzatvorený
```

8.10. Rozšírenie rozhrania `D2SessionEventsListener`

Rozhranie bolo rozšírené o 2 nové metódy:

`void onTerminateRequest()` – je zavolaná v prípade, že niekto prostredníctvom systému D2000 žiada o ukonenie procesu aktuálnej JAPI Session. Napríklad cez TELL príkaz. Korektná implementácia metódy zariadi, aby došlo k ukoneniu zdrojov a následne zavolá `session.close()`.

`void onRedundancyStateChanged(RedundancyStateType redundancyState)` – je zavolaná v prípade prepnutia redundancie D2000 kernela. Hodnota parametra signalizuje aktuálny stav kernela, ku ktorému je pripojená aktuálna session. Korektná implementácia zabezpečí, že sa na komunikáciu so systémom D2000 bude používať session, ktorá je pripojená na HOT kernel.

i Pri migrácii je potrebné všetky implementácie rozšíriť o príslušné metódy.

8.11. Volanie RPC implementujúcej ESL rozhranie

Od verzie 10.1.39 pribudla možnosť vola RPC, ktorá je implementáciou ESL rozhrania. Slúžia na to preaznené metódy `D2Session.callRPC` a `D2Session.callRPCNoReply` s parametrom `interfaceHobj`.

8.12. Test existencie RPC a SBA volania

Vzhľadom k dynamickej povahe konfigurácie D2000 je od verzie 10.1.39 možné za behu overiť, či sa predpokladaná RPC a SBA nachádza na predpokladanom mieste. Slúžia na to metódy `D2Session.testRPC` a `D2Session.testSBA`.

9.13. Informácie o prihlásenom používateľovi a zmena hesla používateľa

Aby klient knižnice JAPI mohol komunikovať s D2000 kernelom, je potrebné, aby mal k dispozícii session, v ktorej je prihlásený platným D2000 používateľským kontom. Pri prihlasovaní sa cez JAPI podliehajú používateľské kontá rovnakým pravidlám, ako pri prihlasovaní sa cez iné D2000 procesy. Napríklad povinnosti pravidelne meniť svoje prihlasovacie heslo. Z toho dôvodu od verzie 10.1.39 pribudlo volanie `D2Session.getLoggedUserInformation`, ktoré sprístupní informácie o prihlásenom používateľovi a stave jeho prihlasovacieho hesla.

Pre možnosť zmeny prihlasovacieho hesla prihláseného používateľa pribudla od verzie 10.1.39 metóda `D2Session.changePassword`, ktorá zmenu hesla umožňuje. Nové heslo musí spĺňať všetky pravidlá, ktoré sú pre prihlasovacie heslá v systéme D2000 zapnuté.



Pri migrácii je odporúčané implementovať funkcionality, ktorá skontroluje stav prihlasovacieho hesla. V prípade, že kernel požaduje zmenu hesla, upozorní používateľa na túto skutočnosť a umožní mu heslo zmeniť. V opačnom prípade hrozí zablokovanie používateľského účtu.

9.14. Zmena správania pri neúspešnom vytvorení D2Session

Vytvorenie objektu typu `D2Session` volaním `D2Connector.createSession` môže zlyhať z niekoľkých príčin. Od verzie 10.1.39 sú tieto príčiny lepšie rozlíšené rôznymi typmi výnimiek, ktoré nastanú pri volaní `createSession`, alebo pri volaní `Future.get`.

- Ak inštancia `D2Connector`-a už nemá aktívne spojenie: `IllegalStateException`
- Ak kernel vyhodnotí parametre `sessionName` alebo `hostName` ako invalidné: `ConnectSessionException`
- Ak kernel odmietne prihlásiť session s uvedenými prihlasovacími údajmi: `LoginFailedException`
 - Ak bolo prihlasovacie meno správne, obsahuje výnimka `PasswordInformation`, v ktorom je uvedený dôvod odmietnutia.
 - V opačnom prípade je dôvod `null`.



Pri migrácii je odporúčané obaliť získanie session do `try ... catch ...` so zachytením príslušných výnimiek.

9.15. Volanie `D2Session.getObjectInfo`

Volanie `D2Session.getObjectInfo` umožňuje získať prístup ku konfigurácii objektu, ktorá je užitočná zvlášť pri zobrazovaní hodnoty objektu používateľovi. V prípade štruktúrovanej premennej s hodnotami získanými z odkazov na iné objekty je potrebné nasledovať „indirect“ odkazy. Od verzie 10.1.39 je k dispozícii volanie `D2Session.getIndirectInfo`, ktoré požiadala kernel o dohadanie informácií na jedno zavolanie.

9.16. Volanie `D2Session.getObjectInfo` vráti aktuálnu hodnotu objektu

Od verzie 10.1.39 je súčasťou výsledku volania `D2Session.getObjectInfo` aj aktuálna hodnota objektu.

9.17. Volanie `D2Session.openObject` vráti informácie o objekte

Od verzie 10.1.39 sú súčasťou výsledku volania `D2Session.openObject` aj informácie o objekte.

9.18. Zmena rozhrania `RPCReturnContext`

Inštancie objektu `RPCReturnContext` sa používajú na zaslanie návratovej hodnoty prichádzajúceho RPC volania (keď ESL zavolá CALL RPC do JAPI). Vo verzii 10.1.39 bolo jej rozhranie a logika ovládania zásadne prepracovaná.

- Metóda `doReturnError` berie ako parameter hodnotu vymenovaného typu `RpcReturnContextErrorType` namiesto číselného kódu (ktorý nebol dokumentovaný). Použitím metódy sa signalizuje, že RPC bola zavolaná nesprávnym spôsobom – nesprávny počet alebo typy parametrov volania.

- Pribudla metóda `doReturnException` ktorou sa signalizuje, že počas vykonávania RPC nastala výnimka.
- Metóda `doReturn`, ktorej parametrom bolo pole hodnôt typu `UnivalValue` zanikla.
- Nahrádza ju metóda `getResponseDataSet`, ktorá sprístupní inštanciu typu `RpcResponseDataSet`.

Vo verzii 10.0.37 bola návratová hodnota zasielaná ako pole hodnôt typu `UnivalValue`. Nevýhodou tohto prístupu bolo, že dďaka poa ani typy jednotlivých `Unival` hodnôt v poli neboli nijak kontrolované, ale ESL interpretér pritom spoľiehal na ich korektnosť.

Z toho dôvodu sa od verzie 10.1.39 návratová hodnota zadáva do objektu typu `RpcResponseDataSet`, ktorý kontroluje typy hodnôt ako aj ich počet.

 Pri migrácii je potrebné upraviť volanie `RPCReturnContext.doReturn(values)` na `RPCReturnContext.getResponseDataSet().setValues(values)`, pričom je pred tým vhodné skontrolovať, či je zadané pole v premennej `values` správnej veľkosti a obsahuje hodnoty požadovaných typov. Taktiež je potrebné upraviť parameter volania `RPCReturnContext.doReturnError` z čísla na hodnoty vymenovaného typu `RpcReturnContextErrorType`.

9.19. Zmena rozhrania `SBAReturnContext`

Inštalácie objektu `SBAReturnContext` sa používajú na zaslanie návratovej hodnoty prichádzajúceho SBA volania z *D2000 Java Runtime* do JAPI:

- Metóda `doReturnError` berie ako parameter hodnotu vymenovaného typu `SBAErrorCode` namiesto číselného kódu (ktorý nebol dokumentovaný). Použitím metódy sa signalizuje, že SBA bola zavolaná nesprávnym spôsobom a nebude ani vykonaná.
- Pribudla metóda `doReturnException` ktorou sa signalizuje, že počas vykonávania SBA nastala výnimka.

 Pri migrácii je potrebné upraviť parameter volania `SBAReturnContext.doReturnError` z číselnej hodnoty na hodnotu vymenovaného typu `SBAErrorCode`.

9.20. Zastaraná `D2SessionUtils.getObjectHOBjByName`

Od verzie 10.1.39 bola zavedená metóda `D2SessionUtils.getFutureObjectHobjByName`, ktorá namiesto hotového výsledku vráti `FutureEvent`, ktorého výsledok je požadované HOBj. Typicky je potrebné získať viac HOBj a použiť ich spolu. Pôvodná metóda vrátila výsledok až keď získala odpoveď z *Kernel-a*, o známe spomaľuje beh programu. Nová metóda umožňuje rýchlo zaslať viac žiadostí a pokiaľ na odpovede spolu.

 Pri migrácii je odporúčané upraviť kód na používanie novej metódy.