Problémy distribuovaného zpracování v OO databázi

Radim Krampol

1. Úvod

Rozšíření objektově orientovaných (OO) jazyků a prostředků při vytváření softwarových produktů je obrovský. Fundamentální myšlenkou OO přístupu je vysoká úroveň abstrakce. OO prostředky tedy modelujeme svět pomocí objektů, vazeb a událostí mezi nimi a využíváme prostředky objektově orientovaného návrhu jako zapouzdření, přísná typovost (třída jako typ objektu), datová nezávislost (odstínění vnitřní struktury pomocí metod), dědění (přirozený vztah evoluce). V této práci se chci věnovat adaptaci myšlenek OO projektování v databázových systémech především distribuovanému zpracování.

Návrh je ovlivněn specifickými požadavky na konečnou aplikaci a našimi zkušenosti z dlouholetého vývoje ekonomických aplikací a informačního systému ve firmě VEMA Brno. Konečná aplikace vytvořená tímto systémem nebude pro jednoho zákazníka nebo skupinu zákazníků stejného typu, ale pro stovky různých organizací z nejrůznějších rezortů a oblastí. To ovlivňuje důraz na modifikaci aplikace v době instalace, to znamená instalování pouze její části, upravenou pro specifika zákazníka. Velký důraz to také klade na konfigurovatelnost.

2. Základní definice a pojetí

Základním stavebním prvkem OO systému je **objekt**. Jeho typ určuje **třída objektu**, charakterizovaná internimi daty a množinou funkcí (**metod**), které lze nad objektem provést. Nad objektem se uplatňuje princip **zapoudření**, to znamená, že vnitřní struktura objektu je uživateli částečně skrytá a on z celým objektem komunikuje pomocí zveřejněného rozhraní. Dalším principem je **dědění**. Obvykle je dědění rozšířením objektu o nové položky a metody či nové definice metod. V našem pojetí tento princip doplňujeme o **zúžení**. Zúžení znamenám omezení oborů hodnot některých položek objektu (někdy se nahradí dokonce konstantou). Objekt je pro objektově orientovaný databázový systém elementárním prvkem, tedy i nejmenší jednotka pro sdílení a distribuování v distribuované instalaci.

Novým prvkem je pojem **modulu**. Modul je množina objektů, které mají mezi sebou z aplikačního hlediska elementární vztah. Jednotlivé moduly mají svoje rozhraní, kterým zpřístupňují svůj lokální svět okolí. Jednotlivé moduly používají objekty jiných modulů, a to buď přímo nebo jako abstraktní objekty, to znamená že i bez znalosti jejich rozhraní (což má význam především při vývoji aplikací). Modul má svůj přímý obraz v modulu definičního a manipulačního jazyka a v metadatech aplikace tj. katalogu.

Modul

- Jazykové části
  - Definice tj. rozhraní
  - Implementační část
  - Vizualizační část
  - Dokumentační část
- Vnějazykové definice

3. Pojem projekt

Nad moduly zavádíme pojem **projekt**. Projekt je množina modulů, které mají z aplikačního hlediska úplný vztah, to znamená, že všechny objekty jsou neabstraktní a tvoří spolu provázaný aplikační celek. Moduly jsou disjunktní navzájem propojené entity. Můžeme tedy konstatovat, že definice objektů tvoří typovou bázi projektů a vazba mezi nimi je popsána v databázi katalogu. Pojem projektu je potom výčet typového jádra projektu (základní moduly) a uzávěr nad ním. Průnik jednotlivých projektů nám určuje vazbu mezi projekty, základní složku mezi projektové komunikace.

Příklad vztahu mezi objektem, modulem a projektem.
(Vztah mezi objekty je vyznačen čarou.)

Modul1
Objekt A1
Objekt B1

Modul2
Objekt A2
Objekt B2

Modul3
Objekt A3

Projekt1
jádro: Modul1
uzávěr: Modul1, Modul2, Modul3

Projekt2:
jádro: Modul3
uzávěr: Modul1, Modul3

Průnikem projektů Projekt1 a Projekt2 je Objekt B1 z modulu Modul1, který se mezi nimi bude sdílet.

4. Typy databází

Pro ujasnění nadprojektového pohledu na OODB G2 rozdělíme aplikaci na jednotlivé typy databází, jak se budou nacházet u koncového uživatele.

- **katalog**
- **administrátorská databáze**
- **datová prostředí**

Katalog je databáze objektů statického popisu aplikace. Jsou v něm uloženy metadata všech projektů, které jsou obsaženy v jedné aplikaci. Při instalaci se pro zákazníka uvolní projekty, které si zakoupil a pomocí projektových uzávěrů další sdílené moduly a objekty z nepoužívaných projektů. Katalog je rozdělen na několik segmentů. V prvním segmentu, který je pro koncového uživatele pouze pro čtení je uložen popis aplikace, jak byl vytvořen programátorem. Další segmety tento popis rozšiřují pomocí standardních prostředků OODB G2 (děděním nebo zúžením). V prvním segmentu je uložena informace o verzi od dodavatele aplikace. Při instalaci vyšší verze (upgrade) je hlavní práce při změně katalogu provedena právě nad tímto segmentem.

Poznámka:

Myšlenka rozdělení katalogu do segmentů mj. umožňuje lokální uživatelské změny do svého segmentu katalogu, který vidí pouze on sám. Tento přístup má velký význam při společné práci

týmu programátorů na jedné aplikaci. Po doladění přídavného segmentu se přidá tato vrstva ostatním programátorů. Je pochopitelné, že z výkonnostních důvodů je počet segmentů omezen. Procházení příliš četných vrstev segmentů i při maximální optimalizaci přístupu znamenalo velké výkonnostní propady.

Administrátorská databáze obsahuje objekty dynamického popisu aplikace nebo přesněji výskytu aplikace. Tady se řeší problémy jako technické nastavení (nastavení terminálů, komunikačních spojení apod.), nastavení prostředí, správa a indentifikace uživatelů, rozdělení databáze do lokalit apod. Na rozdíl od katalogu databáze, který se mění pouze v diskrétních od sebe dostatečně vzdálených okamžicích, administrátorská databáze obsahuje jednak data podobné časové stálosti jako katalog např. typ lokální sítě nebo uživatelská práva v databázi, jednak data, kde si ukládá informace **session manager**, tj. proces který obsluhuje konkrétní uživatelské připojení k databázi

Datová prostředí jsou konkrétní výskyty OO databáze s objekty reprezentujícími uživatelovi data.

Obr. Typy databází

PROJEKTY U ZÁKAZNÍKA (konkrétní výskyt)

KATALOG (statický popis aplikace)

ADMIN (dynamický popis aplikace)

5. Distribuovanost

Požadavkem na distribuované zpracování je, aby rozdíl mezi zpracováním lokální nebo distribuované databáze byl zcela skryt, a to i z hlediska koncového uživatele hotových projektů, tak i z pohledu programátora v OO databázovém systému. Elemetární jednotkou, nad níž lze provádět v OO databázi úvahy o distribuovanosti je objekt.

5.1. Distribuovanost katalogu

Zřejmými entitami distribuce jsou objekty datového prostředí OO databáze. Podle principu ortogonality, který je pro dobré databázové systémy typická, se na katalog, jako metadata aplikací pohlíží stejným způsobem jako na data a stejným způsobem se s nimi pracuje. Pro distribuci jejích objektů lze použít stejné metody jako pro objekty datového prostředí.

Katalog lze mít uložený v distribuovaném systému různými způsoby (podle [2]):

1. Centrální - je uložen právě na jednom centrálním místě
2. Replikovaný - celý katalog je uložen v každém bodu sítě
3. Rozdělený - každý bod sítě má svůj vlastní katalog.
4. Kombinace 1. a 3.

Po analýze jsme se rozhodli pro jednu z variant bodu 4. Dovedli nás k tomu výkonnostní důvody, což je důvod pro rozdělený způsob, A problematika úpravy katalogu v různých místech sítě, což vyžaduje distribuci katalogu spojenou s centrální správou. Nejjednodušší případ centrálního katalogu není vhodný, protože v primárním návrhu uvažujeme i o vzdálených propojeních. Tady se pochopitelně naruší princip ortogonality a zakrytí distribuovaného

přístupu, protože nelze udělat např. iteraci přes distribuovanou databázi, jejíž některé části jsou přístupné modemem, informace z nich přenáší disketami nebo leží na archivních mediích. K pochopení je třeba popsat postup práce při instalaci distribuované databáze. Nejdříve se provede instalace v centrálním místě, kde se provedou instalační úpravy (uživatelské uzpůsobení), doplní se omezení možnosti změny určitých katalogových objektů (obyčejně těch sdílených) a prohlásí se za modifikovatelné centrálně. Každá změna v centru vede k distribuci do jednotlivých bodů sítě, které tuto změnu nemají možnost provést u sebe. Distribucí všech změn z centra je udržena konzistence katalogu. Tento přístup se zdá být dostatečný pro většinu problémů. Protože změna v katalogu se dá provést pouze standardními mechanismy OO databáze G2, které jsou vždy rozšířením třídy objektu, lze při distribuovaném přenosu snadno zajistit převzetí objektu i pokud neznám jeho úplný popis, stačí mi jen informace o jeho typu, jež nese i informaci z jakého objektu byl vyděděn, případně pokud k tomu mám příležitost může se o jeho popis požádat služba katalogu centrální databáze. Tento mechanismus je umožněn přísnou typovostí objektově orientovaného přístupu, když pro každý objekt, s nímž se pracuje, je znám jeho typ. Opačný případ kdy je poslán objekt, jehož typ je starý, umožňuje buď odmítnutí tohoto objektu a počkání na objekt správné verze z místa vzniku nebo jeho přijetí a doplnění "šedých hodnot" či implicitních hodnot do případných neexistujících položek. Mechanismus "šedých hodnot" je možnost definovat pro každou dosud nedefinovanou položku podle jejího způsobu vzniku, při vzniku či změně objektu. Umožňuje systémovou inicializaci hodnot objektu, což může výrazně pomoci programátorovi v G2 zjednodušit velké skupině uživatelů práci s některými objekty.

5.2. Distribuovanost administrátorské databáze

Administrátorská databáze není zcela čistě databází, je to doplnění statického popisu aplikace o popis dynamický popis nasazení aplikace u uživatele a případně některé procesy právy této databáze. Tato databáze obsahuje popis topologie instalace, jako nejvyšší rozdělení databáze, tzn. výskyty jednotlivých částí distribuované databáze a způsob jejich propojení. Důležitým prvkem je údržba databáze uživatelů. Fundamentem našeho pojetí databáze uživatelů je sdružení uživatelů do skupin, při čemž členem skupiny může být nejen uživatel, ale i skupina. Výsledkem je acyklický orientovaný graf (z praktického hlediska je vhodné se omezit na množinu stromů), jehož uzly jsou skupiny a koncové listy jsou buď uživatelé nebo skupiny. Celý graf je atributován nastaveními (barvy, přístupová práva, spuštění a konfigurace tiskáren apod.) a dodatečnou informací o povinném či doporučeném dědění pro podřízené (ve směru orientace hran) uzly grafu. Administrátorská databáze obsahuje popis tohoto grafu ve třech částech; popis voleb nastavení a jejich hodnoty (atributy), popis členství ve skupinách uživatelů (topologie) a vztah mezi topologií a atributy. Pomocí tohoto mechanismu je celá atabáze rozdělena na extenty (jeden atribut skupiny objektů), což je umožněno tím, že při instalaci pro sdílené používání databáze (víceuživatelský přístup) se odkryje systémový atribut objektu - příslušnst ke skupině objektů. Zatímco identifikace objektů pomocí OID je udržována v nižších vrstvách databázového systému, tyto skupiny objektů musí administrátorská databáze udržovat unikátně po celé distribuované síti. Parametrem skupiny objektů se určují přístupová práva k objektům databáze. Tyto disjunktní množiny jsou entitami, mezi nimiž bude probíhat distribuce objektů.

5.3. Distribuovanost datových prostředí

Pro datová prostředí se opakují základní principy předchozích databází. Distribuované části databáze tvoří jednu databázi a uživatel pracuje s distribuovanými prostředky stejně jako se svými lokálními. Jednotlivé objekty jsou rozděleny na disjunktní množiny místem svého fyzického výskytu a rozdělením do skupin objektů. To znamená, že používáme stejného

principu pro distribuci objektů mezi jednotlivými databázemi sítě i mezi skupinami objektů v jedné lokální databázi.

6. Mechanismus front objektů

Pro přesun informace jsme zvolili mechanismus pošty. Pro každou lokalitu nebo skupinu objektů je určena poštovní schránka, což je fronta objektů, které jsou určeny pro přesun mezi místy výměny.

Podle tohoto hlediska jsem rozdělil práci s objekty do 3 typů:
- distribuovaně sdílený objekt
- přesun objetu
- štěpení objektu

Tyto pojmy jsou dostatečně jasné. Jen pro přesun objektu bych poznamenal, že aplikovaný v lokální databázi znamená pro práci s objektem změnu viditelnosti.

Impulsy přesunu jsem rozdělili o těchto kategorii:
- událost typu změna dat (update)
- manuálně (interaktivně řízený přesun nebo dávkově)
- časová událost

Dalším atributem položky ve frontě je její adresa. ta se odkazuje na údaje v administrátorské databázi. Podle místa určení mohou být typu:
- 1:1
- 1:m

7. Sdíleni objektů a verzování

Pro sdíleni objektů lze v administrátorské databázi doplnit údaj o referenčním místě objektu nebo skupiny objektů. Při změně centrálního objektu se tento objekt vyjme z centra a po změně se vrátí zpět, byl-li mezi tím jiný požadavek na tento objekt, pak je podle typu operace a objektu tento požadavek nevyřízen (řídce) nebo se mu podá stará verze objektu. V tom případě vzniká v centru větvení objektů (podverze), které se později opět musí sjednotit ve vyšší celou verzi objektu. Stane se tak obvykle manuálním zásahem uživatele s dostatečnými přístupovými právy. Do doby vyšší verze se pracuje s některou nižší verzí a po vzniku vyšší verze se tato změna distribuuje.

Literatura:

[1] "The Object Database Standard ODMG-93, Release 1.1", Morgan Kaufmann Publishers (1994)
[2] Date,C.J.:"An introduction to database systems", Addison-Wesley (1995)
[3] Grady Booch: "Object Oriented Design With Applications", The Benjamin/Cummings Company Inc.(1991)

ing. Radim Krampol
Škroupova 3
636 00 BRNO
tel. +5-538575