SQL Processor Framework

SQL Processor Framework (dále jen SQLPF) je nástroj pro programové vytváření a spouštění databázových příkazů nad relační databází. Představuje tedy rozhraní datové vrstvy, která se stará o perzistentní uložení dat a ke které přistupuje aplikační (resp. prezentační) vrstva klasické třívrstvé softwarové architektury. SQLPF tvoří alternativu k běžnějším frameworkům pro objektově-relační mapování (ORM), které jsou založeny na budování složitého objektového modelu a reprezentací objektů aplikační či prezentační vrstvy v relační databázi.

Oproti ORM přístupu využívá SQLPF doménově-specifické jazyky pro popis vkládaných či vyhledávaných dat, pro definování „šablon" SQL dotazů a pro výstupní mapování dat získaných z databáze. Tento přístup by měl přinášet především tyto výhody:

- Omezení množství režijního kódu, který je potřeba v aplikační a prezentační vrstvě implementovat. Vkládaná a vyhledávaná data představují jednoduché objekty, které nemusí být součástí složitých objektových modelů nebo frameworků.
- Vysoký výkon datové vrstvy, který není brzděn mnoha úrovněmi abstrakce a umožňuje snadno vytvářet aplikace s odezvou v řádu desítek milisekund.
- Popis rozhraní mezi aplikační/prezentační a perzistentní vrstvou přehlednými a intuitivními prostředky (vhodně navržené doménově-specifické jazyky).

Tento softwarový projekt si klade tyto tři základní cíle:

- Sjednocení a vylepšení gramatik doménově-specifických jazyků použitých v projektu SQLPF.
- Rozšíření implementace SQLPF o nové vlastnosti.
- Refaktorizace implementace s použitím novějších nástrojů pro vývoj doménově-specifických jazyků, modularizace implementace, zvýšení pokrytí kódu jednotkovými testy a rozšíření možností nasazení projektu.

Kontext projektu

SQLPF je dlouhodobě vyvíjen ve společnosti I.CZ jako produkční nástroj pro vytváření rozsáhlých vícevrstvých aplikací v Javě, na které jsou současně kladeny netriviální nároky jak z hlediska výkonu, tak z hlediska spolehlivosti. Mezi důvody pro jeho vývoj patří především eliminace složitých ORM frameworků typu Hibernate pro přístup k databázi, možnost získat plnou kontrolu nad prováděnými SQL příkazy (včetně efektivního využití uložených procedur a podobných pokročilých vlastností konkrétních moderních relačních databází) a zároveň použití intuitivních a snadno použitelných doménově-specifických jazyků pro modelování datových objektů (na rozdíl od užití obecného UML, jež obvykle vyžaduje drahé proprietární UML nástroje).

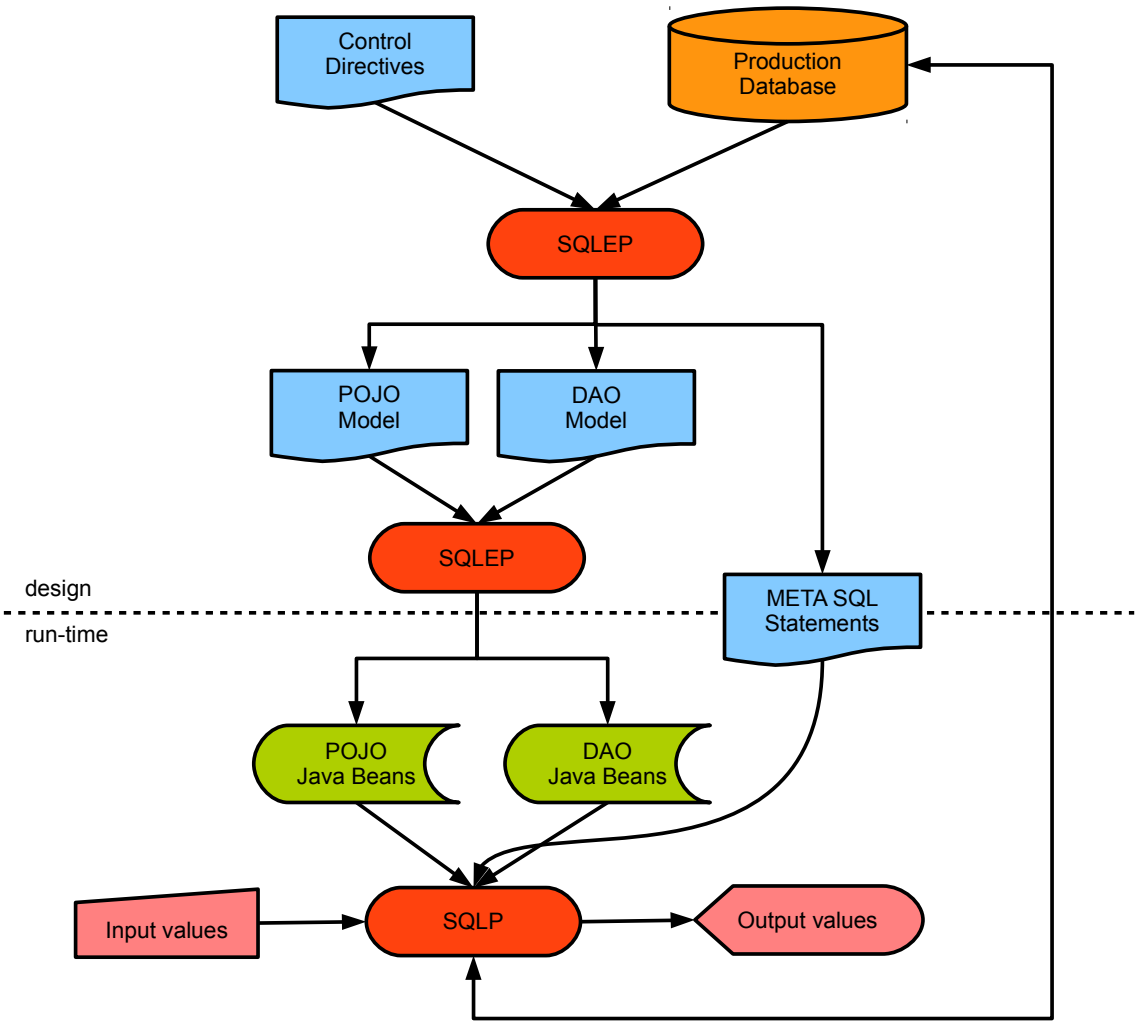
Svou filozofií se SQLPF blíží například frameworku MyBatis, který ovšem místo doménově-specifických jazyků využívá XML.

Architektura projektu

Projekt SQLPF je logicky rozdělen na dvě části:

- **Run-time (*SQL Processor*, SQLP [2]):** Samotné výkonné jádro pro transparentní obsluhu SQL příkazů v aplikačním prostředí. Toto jádro je optimalizováno na rychlost běhu v cílové aplikaci. V jádru se používá META SQL jazyk (založený na gramatice ANTLR [4]), který mapuje Java objekty na konkrétní artefakty a konstrukty v jazyce SQL (vstupní/výstupní atributy apod.).
- **Modelovací nástroj (*SQL Processor Eclipse Plugin*, SQLEP [3]):** Tento nástroj slouží ke generování POJO/DAO Java tříd a META SQL příkazů na základě „reverse engineeringu" databázového schématu. SQLEP využívá Eclipse Modeling Framework a artefakty popisuje doménově-specifickými jazyky postavanými na Xtext [5].

Projekt je šířen jako open source na GitHubu. Část SQLP je šířena pod licencí LGPL a část SQLEP je šířena pod licencí EPL.



Typický proces použití SQLPF uživatelem/programátorem je následující:

1. SQLEP načte databázové schéma a analyzuje relevantní databázové objekty (tabulky, pohledy, procedury, funkce, indexy atd.).
2. SQLEP namapuje relační databázi na objektové soubory, vytvoří POJO/DAO model a META SQL příkazy. Toto mapování může být jak automatické, tak manuální řízené uživatelem.
3. SQLEP vygeneruje příslušné POJO/DAO třídy, optimalizované na efektivní a rychlé zpracování.

Typické použití SQLPF v cílové aplikaci je následující:

1. SQLP naparsuje META SQL příkazy do vnitřní reprezentace optimalizované na efektivní a rychlé použití.
2. SQLP na základě vstupních hodnot aplikační vrstvy (POJO objektů) generuje z předzpracovaných META SQL příkazů finální SQL příkazy, ty pomocí JDBC obslouží a případné výstupní hodnoty naplní opět do požadovaných POJO objektů. Kromě toho SQLP realizuje další QoS služby (monitoring, statistiky provozu, stránkování, transparentní obsluha sekvencí a identit atd.).

Současná omezení

Historicky nejstarší část SQLP je META SQL jazyk, jehož gramatika je znakově orientovaná. META SQL příkazy jsou vlastně znakové řetězce, ve kterých SQLP identifikuje šablony a vzory. Tyto šablony a vzory pak umožňují dosazovat vstupní parametry do generovaných SQL příkazů a/nebo reprezentovat výstupní parametry. SQLEP lze poté použít pro interaktivní validaci správného užití vstupních a výstupních atributů.

Naproti tomu gramatika pro psaní kontrolních direktiv a POJO/DAO modelování je objektově orientovaná. Z toho plyne základní rozpor mezi doménově-specifickými jazyky obou částí projektu. Současná syntaxe objektové gramatiky je také velice proprietární z důvodu nutnosti koexistence se znakově orientovanou gramatikou a je tudíž konceptuálně poměrně vzdálená jazyku Java.

Projektový tým

Vedoucí: Martin Děcký, MFF UK (decky@d3s.mff.cuni.cz)

Konzultant: Vladimír Hudec, I.CZ (vladimir.hudec@i.cz)

Řešitelský tým:

- TBD (ideálně 6 členů)

Zadání projektu

1. Zlepšení modularity kódu SQLPF, tj. především vytvoření samostatných modulů SQLP a SQLEP.
2. Doplnění možnosti variantního překladu projektu pomocí technologií Maven a Tycho.
3. Doplnění jednotkových testů pro SQLEP.
4. Návrh vhodného použití a využití nástrojů Xbase [6], Xtext [5] a Xtend [7] pro definování doménově-specifických jazyků v SQLPF.
    - Sjednocení gramatik doménově-specifických jazyků v SQLP a SQLEP tak, aby byly konceptuálně bližší jazyku Java. Pro toto sjednocení gramatik je nutné sestavit katalog všech sémantických funkcí SQLEP a pro tuto funkcionalitu je poté nutné hledat vhodné syntaktické vyjádření.
    - Rozdělení generování Java kódu do několika Xtend modulů podle typu cílového objektu (POJO, DAO).

Úspěšnost splnění zadání je možné vyhodnotit kromě objektivních kritérií také subjektivním zjednodušením syntaxe použitého doménově-specifického jazyka. Například stávající syntaxe pro označení, že entita *Contact* je serializovatelná se *serialVersionUID=1* a s primárním klíčem *id*, který je současně používaným indexem, vypadá následovně:

```
pojo Contact serializable 1 {
	id : java.lang.Long primaryKey index 1
	...
}
```

Nová syntaxe konceptuálně bližší jazyku Java může vypadat například následovně:

```
@Serializable(1)
@Pojo
Contact {
	@PrimaryKey
	@Index(1)
	id : java.lang.Long
	...
}
```

Analýza náročnosti a rizik

Projekt klade na řešitele především tyto nároky:

- Detailní poznání architektury a implementace projektu SQLPF (v aktuálním stavu). *[1 měsíc]*
- Nastudování potřebných technologií a nástrojů, viz literatura. *[2 měsíce]*
- Implementace přímočarých bodů zadání (1. až 3.). *[1 měsíc]*
- Analýza a implementace komplexního bodu zadání (4.). *[4 měsíce]*
- Ladění, dokumentace. *[1 měsíc]*

Projekt SQLPF staví v základu na stabilních a prověřených technologiích jako je Java a ANSI SQL. Také další technologie a nástroje, které tento projekt navrhuje využít (Xtext, Xtend atd.), lze považovat za dostatečně vyzrálé, přestože se neustále vyvíjejí.

Projekt SQLPF je aktivně vyvíjen, udržován a využíván společností I.CZ. Tato společnost má tedy zjevný zájem na tom, aby byl tento softwarový projekt úspěšný. Hlavní architekt a vývojář SQLPF ze společnosti I.CZ bude působit jako konzultant projektového týmu a bude aktivně mentorovat členy řešitelského týmu.

Největší riziko pro úspěšné dokončení projektu lze spatřovat především v kombinaci několika netriviálních technologií a nástrojů, které musí řešitelský tým dobře zvládnout. Z tohoto důvodu se předpokládá, že se řešitelský tým bude celou 1/3 doby běhu projektu seznamovat s potřebnými technologiemi a nástroji. Současně budou řešitelé po celou dobu běhu projektu v úzkém kontaktu s hlavním architektem a vývojářem SQLPF.

Lze říci, že celková náročnost a rizika projektu odpovídají očekávanému workloadu pro šestičlenný řešitelský tým a devítiměsíční dobu vypracování.

Literatura a reference

Úspěšnost projektu je podmíněna správným pochopením několika netriviálních technologií. Z tohoto pohledu lze především knihu [1] považovat za povinnou četbu pro členy řešitelského týmu.

[1] Lorenzo Bettini: *Implementing Domain-Specific Languages with Xtext and Xtend*, https://www.packtpub.com/application-development/implementing-domain-specific-languages-xtext-and-xtend

[2] SQLP, https://github.com/hudec/sql-processor/wiki

[3] SQLEP, https://github.com/hudec/sql-processor-eclipse/wiki

[4] ANTLR, http://www.antlr.org/

[5] Xtext, http://www.eclipse.org/Xtext/

[6] Xbase, http://www.eclipse.org/Xtext/documentation.html#xbaseJavaReferences

[7] Xtend, http://www.eclipse.org/xtend/