|  |  |
| --- | --- |
|  | ВЕЛИКОТЪРНОВСКИ УНИВЕРСИТЕТ  “СВ. СВ. КИРИЛ И МЕТОДИЙ” |

Владимир Йорданов Йотов

Модели, базирани на йерархични композиции от пространства, за управление на софтуерни версии

Автореферат

на ДИСЕРТАЦИЯ

за присъждане на образователна и научна степен “Доктор”

по научна специалност: 01.01.12 “Информатика”

Научен ръководител: доц. д-р Хр. Тужаров

Велико Търново  
2012

# Въведение

## Актуалност

Управлението на версията на софтуерните продукти заема важно място в областта на софтуерното инжинерство [G365, G100]. Въпреки наличието на разработени модели, научно-приложната област предоставя възможности за търсене на решения за постигане на по-висока ефективност на работният процес. Модерните гъвкави методологии предлагат един по-свободен начин на развитие на софтуерните продукти, като те предполагат използването на специалисти от високо ниво, познаващи разработваният продукт и използваната технология в детайли. Създаването и задържането на екипи от такива специалисти представлява предизвикателство за всяко едно корпоративно ръководството. Настоящата дисертация може да се разглежда като опит да се предостави възможност за снижаване на рисковете от използването на такива кадри и въвеждането на инструментариум за автоматизация при използването на гъвкавите методологии.

## Обект и методология на изследването

Обект на изследване на дисертацията са моделите и методите в управлението на версии чрез използването на йерархично композирани работни пространства за постигане на по-ефективен подход на нейното управление, ускоряване анализа на влиянието на промените над системата, усъвършенстване политиката на управление на знания в компаниите, и инструмент за обсъждане финансовите аспекти на проектите.

Методология на изследването включва следните подходи: анализ на целите и задачите, стоящи пред моделите в научно-приложната област; сравнителен анализ на използваните модели и методи и определяне на нови идеи; търсене, изследване и развитие на модели и методи за управление на версия и повишаване ефективността на процеса на създаване и поддържане на софтуерните продукти.

## Цели и задачи дисертацията

**Цел:** Да се изследват, създадат и развият на модели за управление на софтуерни версии в среда, базирана на йерархично композирани на работни пространства.

Във връзка с основните цели се поставят следните задачи:

1. Да се създаде модел на версионизиран обект, осигуряващ максимална гъвкавост при определяне степента на гранулираност на данните в съчетание с простота и универсалност.
2. Да се създаде модел на среда с йерархично композирани работни пространства, както и да се определят правилата за управление на версия на обекти в тази среда.
3. Да се адаптира метод за проследимост на промени, базиран на събития, за среда с модел на йерархично композирани работни пространства.
4. Да се определи терминологията в областта на версионизирането с използването на йерархично композирани работни пространства.
5. Да се създаде методологична рамка за създаване на софтуерни продукти в среда с йерархично композирани работни пространства.
6. Да се докаже повишаване на ефективността при разработването на софтуерни продукти при разработените модели.

## Структура на дисертационния труд

Дисертацията се състои от увод, три глави, заключение, използвана литература, две приложения и прототип.

**В Първа глава** е направен обзор на моделите в областта на управлението на версиите. Разгледани са мястото, целите и задачите на управлението на версии в рамките на разработването и поддържането на софтуерни продукти. Направен е обзор на съществуващите модели на версионизираните обекти и на начинът на тяхното съхраняване в репозиторито с версии. Отделно е направен обзор на темата за съвместната работа на сътрудниците, където е наблегнато на работните пространства като средство за осъществяването на кооперираност. За постигане пълнота на пълнота в обзора са сравнени методите за проследимост на промените. Главата завършва с определяне на изводите, формиране на целта и задачите на дисертацията, които следва да бъдат решени във втора и трета глава.

**Във Втора глава** са представени теоретичните модели за управление на версия в среда с йерархично композирани работни пространства. Моделите са допълнени с авторска методологична рамка за тяхното ефективно използване. Във формулираните в края на главата изводи са посочени предимствата на разработените модели.

**Трета глава** съдържа аналитичено обоснован избор на средства за реализиране на програмен прототип на система, реализираща теоретичните модели. Представени са описания на авторска алгоритмична реализация на по-важните моменти от прототипа. В главата е направена теоретично-експериментална сравнителна симулация на разработка на програмен продукт с и без използване на разработения прототип при гъвкава методология за разработване. Направените изводи в края на главата показват предимствата на разработените модели.

В заключението е обобщение получените резултати. Формулирани са основните резултати в рамката на дисертацията. Посочени са някои актуални задачи, които могат да бъдат естествено продължение на настоящото изследване.

Разработката и апробацията на резултатите са извършени самостоятелно, като регулярно са представяни в катедра „Компютърни технологии” на Великотърновски университет „Св. св. Кирил и Методий”.

# Публикации, свързани с дисертационния труд

1. Jotov, V. An investigation on the approaches for version control systems. In Proceedings of the 9th international Conference on Computer Systems and Technologies and Workshop For PhD Students in Computing (Gabrovo, Bulgaria, June 12 - 13, 2008). B. Rachev and A. Smrikarov, Eds. CompSysTech '08, vol. 374. ACM, New York, NY, V.11-1. DOI= http://doi.acm.org/10.1145/1500879.1500959.
2. Jotov, Vl., Transaction over Versioned Objects in Hierarchical Workspace Environment, ECAI09, 2009, Pitesti, Romania.
3. Jotov, Vl., Towards a model of versioning domain, 5th Central and Eastern European Software Engineering Conference in Russia, Moscow, 28-29 Oct. 2009, ISBN 978-1-4244-5665-9, http://dx.doi.org/10.1109/CEE-SECR.2009.5501147.
4. Йотов, Вл., Модел на данните в система за контрол на версии, базирана на йерархични работни пространства, 6 – 7. 11. 2009г, Велико Търново.
5. Jotov, Vl., Adaptation of Event-Based Traceability Method for Environment with Hierarchal Composed Workspaces, In Proceedings of International Conference Automatics and Informatics’10, Sofia, 3 – 7 October 2010.

# Съдържание на дисертационния труд

## Глава 1. Управление на версията при създаването на софтуерни системи

В първа глава е са разгледани съществуващите модели в областта на управлението на софтуерни версии. В резултат направеният обзор в се налагат следните изводи:

1. Системите за управление и контрол на версии представляват задължителен инфраструктурен инструмент в съвременното софтуерно производство. Промени в моделите на управлениета на версиите играе водеща роля над целият процес на създаване на софтуерни продукти.
2. Направен е анализ на различни модели за представяне и съхраняване на версионизирани обекти. Изтъкнати са предимствата и недостатъците на разгледаните модели. Определена е необходимостта от поддръжката на различно ниво на гранулираност при версионизираните обекти, обусловена от практиката на използване на няколко нива на абстракция при разработката на софтуерни системи. Нива на абстракция успешно може да се съпоставят на нивата на гранулираност.
3. Направен е анализ на различни подходи за съхраняване на версионизирани обекти. Подходът за съхраняване състояния на версионизираните обекти, предполага проста реализация и по-висока скорост системата.
4. Анализирани са предизвикателствата стоящи пред съвместната работа над един продукт. Определена е необходимостта от научно изследване на моделите и механизмите, използвани в областта на йерархично композираните работни пространства. Установено е, че йерархично композираните работни пространства служат като инструмент за осигурявана на автономна работа, като съчетават това възможност за коопериране на работата между участниците в процеса по създаване на софтуерни продукти.
5. Направен е анализ на темата за проследимост на промените. Представени са различните видове проследяващи връзки, както и методите за получаването им.

## Глава 2. Модели за управление на версии в среда с йерархична композиция на работни пространства

### 2.1. Модел на версионизиран обект

Водещите автори в областта на управлението и контрола на версиите [G42, G8] определят версионизираните обекти, като съставен от две части – състояния на обекта (версии) и граф на версиите, където Под граф на версиите се разбира такъв граф, чиито върховете представляват отделните състояния (версии) на обекта, а ребрата съответстват на логическата последователност на създаване на версиите.

Първата нова характеристика, която следва да притежава един модел на версионизиран обект е той да предоставя възможност да се определи нивото на детайлизираност, т.е. на гранулираност, която самият модел следва да поддържа.

1. Съставен обект ще наричаме обект, който е съставен от други обекти (версии на обекти) посредством композиции.
2. Композиция представлява същност определяща връзката между супер-обект и под-обект. Един съставен обект може да бъде супер-обект на една или повече композиции, т.е. да е съставен от един или повече под-обекти.



Фиг. Пример за промяни на съставността на обектите

Същността **версионизиран обект** е необходимо да притежава само и единствено уникален и непроменяем номер, който е удачно да се използва и като първичен ключ за същността. Версиите на един обект може да се разглеждат като негови примитиви (**версионизирани примитиви)**, чиито основни атрибути са:

* Глобален номер на версията
* Номер на версионизиран обект, с който дадената версия е свързан;
* уникален Номер на версия в рамките на обекта.
* Наименование на обекта. Така полученият модел става по-пълноценен, елиминерайки недостатъка свързан с преименоването на обектите (файловете) при системи като CVS, SVN, Git, Metcury и др. [G50, G54, G101, G387, G389].
* Съдържание на обекта, включващи данните в съответната версията на обекта.

Версионизираният примитив се определя еднозначно посредством уникалната двойка **номер на версионизиран обект** и **номер на версия.**

За нуждите на версионизиране на съставни обектите, следва да се дефинира допълнителна същност - „Композиция на версионизирани примитиви” (накратко композиция), която е еднозначно свързва версията на супер обекта с версиите на неговите под-обекти. Атрибутите на същността са:

* Глобален номер на супер-обекта;
* Глобален номер на под-обекта



Фиг. ER модел на версионизиран обект

За нуждите на отчетността и проследимостта на промените, така създаденият модела следва да се разшири с цел да да поддържа граф на верссите. В ER моделите е прието графовата структура да се моделира от две същности – същност на възлите и същност на дъгите [G410].Тук дъгите на графа (показващи прехода от една верися в друга) следва да притежават следните атрибути: Номер на дъгата (първичен ключ); глобален номер на изходната версия; глобален номер на целевата версия; Потребител, извършил промяната; Дата и час на промяната; Допълнителни данни относно промяната.

#### 2.1.1. Версионизиране на съставен версионизиран обект

Настоящата подсекция има за цел да представи особеностите при управлението на версия на съставни обекти от първи ред. Базирайки се на тях, ще се определи процесът на версионизиране на съставни обекти от ред N.

1. Съставен обект от ред **0** ще наричаме такъв обект, за който няма асоциирани под-обекти, т.е. фактически той не се явява съставен обект. Съставен обект от ред **N** ще наричаме такъв обект, за който най-големият ред на асоцииран под-обект е равен на **N-1**.
2. Един под-обект сам по себе си може да се явява съставен обект от други обекти, като по този начин да се създаде суперпозиция от съставни обекти.

Една от основните задачи, която стои пред настоящият научно-приложен труд е да не усложняваме без необходимост тук създадените модели. Изхождайки от това, както и на факта от липсваща практическа необходимост, при построяването на суперпозиция от съставни обекти следва да въведем следното ограничаващи правила (ограничения):

1. В дадена суперпозиция от съставни обекти, обект може да присъства най-много един път.
2. Един обект може да присъства най-много в една суперпозиция от обекти.
3. При промяна на версията на даден под-обект за даден супер-обект, това не влиае на версиите на другите под-обекти, съставящи супер-обекта (Фиг. 32).



Фиг. При промяна във версията на един под-обект, не се променя версията на съседните под-обекти

1. Версия на даден съставен обект е видима в дадено работно пространство, само и единствено, когато всички версии на съставящите го под-обекти са видими в съответното работно пространство.

### 2.2. Йерархично композирани работни пространства. Модел на видимост на версионизирани обекти

#### 2.2.1. Модел на йерархично композирани работни пространства

1. Продукт се нарича обект на материалното или не материалното производство, който след своето създаване може да бъде размножен и разпространяван.
2. Издание на продукт се нарича определена фиксирана негова версия. Само издания на продукта се разпространяват сред клиентите.
3. Работно пространство се нарича място, където се извъриват определени дейности по създаването на версия на продукт.
4. Главно работно пространство се нарича работно пространство, в което се извършва окончателната сборка и подготовка на издание на продукта.



Фиг. Клас диаграма на модел продукт-издение-работно пространств  


Фиг. Примерна йерархична композиция на пространства

#### 2.2.2. Модел на видимост на версионизирани обекти в среда с йерархично композиране на работни пространства

Под видима версия на версионизиран обект ще се разбира версията или липсата на версия за обекта, с която потребителят (разработчикът) работи в рамките на определено пространство, което от своя страна се нарича локално работно пространство. Под локална версия на версионизиран обект за дадено работно пространство, ще се разбираме такава негова версия, която е асоциирана с работното пространство.

Принципи на видимост:

1. Ако даден версионизиран обект има версия в рамките на дадено пространство (локална версия), то в това пространство се вижда само тази версия на обекта, въпреки наличието на други версии в родителските пространства.
2. Локалната версия на обект от дадено работно пространство се вижда рекурсивно във всички под-пространства, освен ако няма дефинирана друга локална версия в тях.

От изложените принципи можем да изведем следствията:

1. В всяко работно пространство, където обектите нямат локална версия, са представени с тяхна версия намираща се в най-близкото родителско работно пространство.
2. Ако за дадено работно пространство обекта няма версия в нито едно родителско работно пространство, то той не се вижда в първоначално избраното работно пространство.



Фиг. Разпределение на версиите на версионизиран обект съгласно принципите на видимост

### 2.3. Транзакции над версионизиран обекти

#### 2.3.1. Транзакции над версионизиран обект в рамките на едно работно пространство

По-долу са разгледани следните транзакции: създаване на версионизиран обект; актуализиране на не локален версионизиран обект; създаване на маркер на състояние (state-mark) на локален версионизиран обект, маркер на състояние изтрит обект и отказ от маркер на състояние.

Създаването е първоначалната транзакция за всеки един версионизиран обект, след изпълнението на която, обектът притежава първоначална (нулева) версия.

Създаването на маркер на състояние представлява транзакция, при която се създава нова версия на даден версионизиран обект.

Като обратна транзакция за създаване на състояние може да се квалифицира тази по отказ от маркер на състояние. Чрез нея в представеният модел последното състояние се освобождава, а текуща локална версия на обекта става версията предхождаща отказаната.

Актуализирането на не-локален версионизиран обект представлява сложна транзакция, която се състои от следните стъпки:

* Извличане на предишната видима за работното пространство версия на обекта;
* Създаване на локална версия на обекта в текущото работно пространство;
* Създаване на релация на версиите (дъга в графа на версиите).

Изтриването на даден обект е възможно чрез транзакция за създаване на т.нар. маркер за изтрит обект.

#### 2.3.2. Транзакции над версионизиран обект между две работни пространства

Транзакциите между две работни пространства може да се разделят на две групи – публикуване на версия на обект и отказ от локална версия. Преди да се разгледат е необходимо да се въведат термините „производна” и „паралелна” (непроизводна) версия на обект.

1. Нека разгледаме един версионизиран обект и две негови версии X и Y. Ако съществува път в графа на весии на обекта от версия Х до версия Y, то версия Y се явява **производна версия** на версия Х, а версия на Х – предшестваща версия Y.
2. Нека разгледаме един версионизиран обект и две негови версии X и Y. Ако не съществува път в графа на версиите за обекта от версия Х до версия Y, то двете версии се явяват **паралелни (непроизводни) версии**.

Под публикуване на версия на обект, ще се разбира поредицата от действия, необходими за привеждане локалната версия на обекта от текущото работно прострастнво в локална версия в родителското работно пространство.

Простото публикуване на версия е при ситуация, когато в родетилското работно пространство не съществува локална версия на публикуваният обект.

За транзакцията на актуализиращо публикуване. Характерно при нея е, че тя е възможна, когато бъдат едновременно изпълнени следните две условия:

* в родителското работно пространство съществува локална версия на обекта, който се публикува;

версията на обекта, който се публикува, се явява производна на версията му в родителското работно пространство.

При актуализиращото публикуване не се извършва сливане между двете версии понеже производната версия представлява еволюционно продължение на предшестващата версия.

Когато версията обекта, която се публикува в родителското работно пространство се явява паралелна спрямо намиращата се там локална версия (Фиг. 38), тогава следва двете версии да се слеят. Като резултат на сливането се получава нова версия на обекта.

Транзакцията по отказ от локална версия явява обратна на транзакциите по публикуване на версия. Тя включва само една стъпка: премахване на локалната версия на обекта от работното пространство. При премахването сработват механизмите от от модела на видимост на версионизираните обекти.



Фиг. Публикуване със сливане

#### 2.3.3. Транзакции над съставни обекти

При ситуацията, когато имаме локална версия на обекта В в родителското работно пространство и негова видима версия в текущото работно пространство. В текущото работно пространство се създава под-обект А за обекта В (Фиг. 39). При публикуването версията на под-обекта А е възможно и тя не води до някаква промяна във версията на обект В в родителското работно пространство. Въпреки това при последващо публикуване версията на обекта В заедно с неговите композиции, в родителското работно пространство ще доведе до автоматично обновяване (в рамките на работното пространство) на композиционната схема на обектите (Фиг. 39 – зелената пунктирана стрелка). Това е продиктувано от факта, че информацията относно организацията на съставният обект следва да се разглежда като неделима част от него.

При публикуване на новата версия на съставният обект (B,v3) води до изискването това да се извърши в комплект с версията на новосъздаденият под-обект (Фиг. 39 – стрелките с №2).



Фиг. Новосъздаен под-обект към супер-обект

1. Публикуването на версия на локален съставен обект следва да се извършва в комплект с всички локални версии на неговите под-обекти, които имат различна версия в родителското работно пространство (Фиг. 40 – зелената и жълтата стрелки с №2).



Фиг. Индиректна променена версия на супер-обект, породена от нова версия на под-обект

1. Публикуването на версия на обект, който притежава предишна версия, явяваща се под-обект на съставен обект в родителското работно пространство на текущото работно пространство, следва да се извършва едновременно с публикуването на локалната версия на съответният съставен обект.
2. Отказът от локална версия на съставен обект следва да се извършва заедно с рекурсивен отказ от локална версия на всички негови под-обекти.

#### 2.3.4. Класификация на транзакциите над версионизирани обекти

1. Транзакции в рамките на едно работно пространство
   1. Прости обекти
      1. Създаване на версионизиран обект
      2. Маркиране на версия
      3. Актуализация на не локален версионизиран обект
      4. Маркиране на изтрито състояние
   2. Съставни обекти
      1. Включване на обект в композицията на съставен обект.
      2. Автоматично регистриране на индиректна нова версия на съставен-обект, породена от нова версия на под-обект.
      3. Изваждане на под-обект от композицията на съставен обект
2. Транзакции между две работни пространства
   1. Прости обекти
      1. Просто публикуване
      2. Актуализиращо публикуване
      3. Публикуване със сливане
      4. Отказ от локална версия
   2. Съставни обекти
      1. Публикуване на съставен обект
      2. Публикуване на обект, изваден от композицията на съставен обект.
      3. Отказ от локална версия

#### 2.3.5. Жизнен цикъл на версионизиран обект



Фиг. Диаграма на състоянията на версионизиран обект

### 2.4. Проследимост на промените в среда с йерархична композиция на работни пространства

## Глава 3. Изследване приложимостта на моделите

...

## Заключение

...

## Използвана литература

...