- розробкою та впровадженням автоматизованої системи управління ресурсами;
- осучаснення матеріально-технічної основи системи управління;
- впровадження в експлуатацію розроблених складових;
- забезпечення комплексного та злагодженого функціонування складових ЄАСУ у єдиному інформаційному середовищі під час виконання завдань Збройними Силами.

Підводячи підсумок, можливо зробити висновок, що проведення заходів зі створення ЄАСУ надасть можливість:

- скоротити час, що витрачається на збирання, оброблення, передавання оперативної інформації та відображення її на автоматизованих робочих місцях службових осіб органів військового управління від найнижчої до найвищої ланки;
- зменшити строки прийняття рішень за рахунок проведення математичного моделювання імовірних дій Збройних Сил та терміни доведення до підпорядкованих військ (сил) завдань, команд, сигналів;
- забезпечити підвищення ступеню реалізації бойових можливостей військ (сил);
- підвищити ефективність роботи командувачів (командирів) і штабів органів управління всіх рівнів та обґрунтованість оперативних (бойових) документів, що розробляються. Проведення математичного моделювання дій військ (сил) надасть їм можливість опрацювати у найкоротший термін кілька варіантів дій та визначити найкращий (оптимальний) за умов обстановки, що складаються;
- привести показники оперативності, стійкості і прихованості управління, як складових ефективності управління, до належного рівня.

На наш погляд, це дозволить підвищити ефективність управління військами (силами) на 45-50 %, що в свою чергу на 15-18% підвищить ефективність їх застосування.

Література

УДК 004.94

ИЗМЕНЧИВОСТЬ В ОБЪЕКТНОЙ МОДЕЛИ САМОВОСПРОИЗВЕДЕНИЯ ФИРМЫ

В.И. Гурьянов
Филиал СПбГИЭУ в г. Чебоксары, Россия

В данном докладе представлена имитационная модель процесса самовоспроизведения, характерная для экономических субъектов, и предложен возможный алгоритм изменчивости, специфический для этого механизма самовоспроизведения. Этот доклад является развитием исследований, представленных на конференции ISDMCI'2012.

В качестве примера механизма самовоспроизведения рассмотрим франчайзинг. Франчайзинг предполагает, что одна сторона (франчайзер) передаёт право на определённый вид бизнеса и проверенную бизнес-систему (франшизу) другой стороне (франчайзи). Франчайзер, как правило, организует жесткий контроль над выполнением стандартов бизнеса. Тем не менее, существует необходимость внесения тех или иных адаптивных изменений в бизнес-систему. Мы рассмотрим адаптацию франшизы под размеры бизнес-системы.

Для описания объектных моделей систем удобно использовать профиль UML, предназначенный для объектно-ориентированного имитационного моделирования [1] (далее – *Scientific Profile*). Профиль представляет собой формальный язык, наделенный двойственной семантикой – предметной и вычислительной. Дальнейшее изложение построено на языке субпрофиля для моделирования активных систем (см.[2]). Языки реализации - Smalltalk и C++.

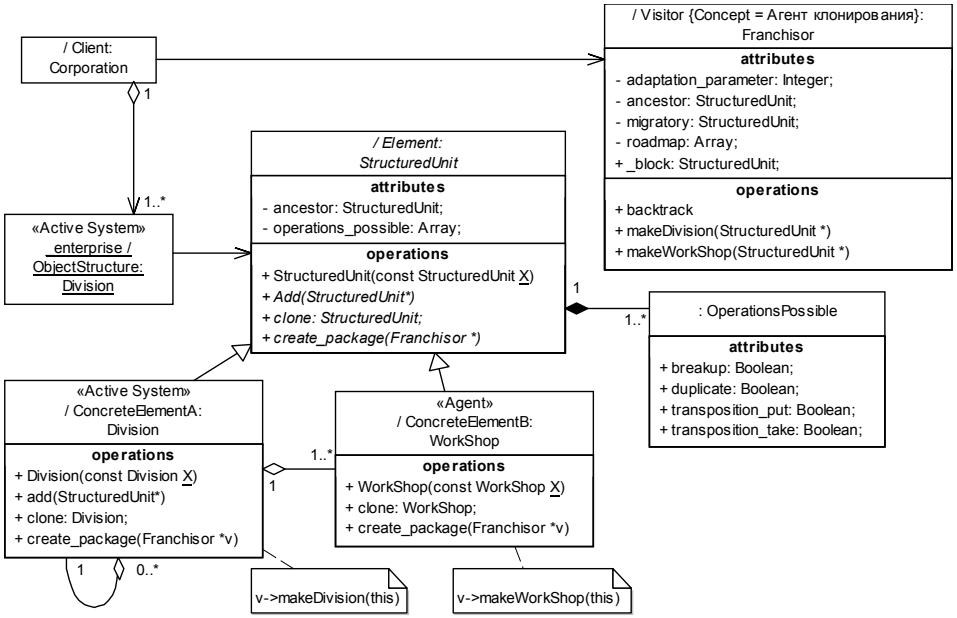

Рисунок 1 - Паттерн *Visitor* для модели самовоспроизведения

Вычислительная семантика. Модель самовоспроизведения фирмы в самом общем случае задается паттерном *Builder* (см. рис.1.). Данная модель есть объектная модель предприятия (паттерн *Composite*; классы StructuredUnit, Division и WorkShop), дополненная свойствами и методами, обеспечивающими процесс самовоспроизведения. Мы ограничимся

«мягким» вариантом паттерна *Builder* – будем полагать, что класс, играющий роль *строителя*, один, но зато он может быть параметризирован. Процедура clone_enterprise класса Corporation{Concept = ИНТЕГРИРОВАННЫЕ БИЗНЕС_ГРУППЫ} создает *агента клонирования*, конфигурирует им метод create_package объекта _enterprise и по завершению процедуры принимает клон бизнес-системы. Вычислительная семантика поведения агента определяется паттерном *Visitor* (см. рис.1); роли классов расписаны согласно [3].

Класс Franchisor (в роли *Visitor*) объявляет и реализует методы makeDivision и makeWorkShop, которые выполняют клонирование текущего элемента и его связей. Агент получает элемент в качестве аргумента и обращается к элементу через интерфейс, посылая сообщение clone. Процесс клонирования описывается паттерном *Prototype*. Кроме того, агент выполняет копирование связей элемента или модифицирует их, работая с полем ancestor. Агент выполняет элементарные операции, руководствуясь переменной operations_possible (см. рис.1) - списком допустимых операций. Переменная задана в каждом элементе структуры и параметрически зависит (зависимость моделируется массивом) от переменной adaptation_parameter, известной агенту. Всего возможно 16 значений переменной operations_possible, однако все они могут быть сведены к следующим: тождественное отображение (¬duplicate ∧ ¬breakup ∧ ¬transposition_take ∧ ¬transposition_put = true), начать перемещение элемента (¬duplicate ∧ breakup ∧ transposition_take ∧ ¬transposition_put = true), начать перемещение дубля (duplicate ∧ ¬breakup ∧ ¬transposition_take ∧ ¬transposition_put = true), завершить перемещение (¬duplicate ∧¬breakup ∧ ¬transposition_take ∧ transposition_put = true), дублировать элемент (duplicate ∧ ¬breakup ∧ ¬transposition_take ∧ transposition_put = true), повторно использовать элемент в техпроцессе (¬duplicate ∧ ¬breakup ∧ transposition_take ∧ transposition_put = true). Для дублирования элемента создается новый экземпляр класса Franchisor. Перемещаемые элементы или их дубли помещаются в поле migratory.

В *Scientific Profile* программный код рассматривается как текст, описывающий предметную область. Каждой программной сущности назначается предметная семантика в виде {Concept = *название концепта*}.

Предметная семантика. Для процессов клонирования характерна высокая устойчивость к изменениям. Основное ограничение - необходимость сохранения технологии бизнес-процессов, - придает дополнительную «жесткость» процессу клонирования.

Концепт «Адаптировать и документировать бизнес-систему» для методов makeDivision и makeWorkShop. Агент клонирования производит документирование и адаптацию бизнес-системы, придерживаясь инструкций, которые могут быть формально описаны следующим образом. Пронумеруем вершины дерева p в порядке их обхода, начиная с нуля. Обо-

значим параметр адаптации как ξ. Зададим в каждой вершине функцию $I(p,\xi)$ (operations_possible), которая определяет список операций, не нарушающих технологию. Введем понятие *квазилокального оператора H над лесом g*: преобразование, порождаемое функцией $I(p,\xi)$ в вершине p, посредством которого лесу g с выделенной вершиной p сопоставляется лес g^*. Функцию $I(p, \xi)$ можно назвать *булевым образом* оператора H для вершины p. Как следует из вышесказанного, можно задать шесть квазилокальных операторов, которые формально будем записывать в следующем виде: $H(p, I(p, \xi)): g \to g^*$.

Концепт «Конфигурирование бизнес-системы». Данный концепт назначается всей программной конструкции, описанной выше. Обозначим суперпозицию операторов, как $H \oplus G$ и будем понимать под этой записью последовательное применение квазилокальных операторов - сначала G, а затем H. Таким образом, процедура клонирования может быть представлена в следующем виде $J(\xi) = \sum H_{n-p}(p, I(p, \xi))$, где p изменяется от 0 до n. Задавая разные функции $I(p,\xi)$ можно определять различные операторы $J(\xi)$, в том числе такие, которые позволяют: (а) переместить элемент, (б) дублировать элемент и переместить дубль. Перемещение элемента структуры возможно как *по*, так и *против* направления обхода структуры, а также на другой уровень.

Для аналитического исследования программной симуляции можно воспользоваться λ-исчислением. Можно показать, что уже во втором поколении клонов имеет место вырождение – происходит упрощение технологического процесса и утрата структурной информации; последнее снижает адаптивность.

Итак, в данном докладе рассмотрен механизм самовоспроизведения, такой, что клон системы создается одновременно с обходом агентом структуры оригинала. Предложены операторы преобразования графа структуры, совместимые с клонированием. Удачные решения фиксируются в структуре фирмы и составляют часть интеллектуальной собственности франчайзера. Тем самым показано, что механизм самовоспроизведения, основанный на клонировании, допускает изменчивость, что обеспечивает адаптацию бизнес-систем.

Литература

1. Гурьянов В.И. Специальный UML-профиль для моделирования сложных систем // Информационные технологии моделирования и управления. – Воронеж: Изд-во «Научная книга», 2010. – № 3(62). – С. 356-362. (см. также http://econf.rae.ru/article/5385)

2. Гурьянов В.И. Логический подход в методологии имитационного моделирования активных систем // ИММОД-2011: труды конференции. – Том 1. – Санкт-Петербург, 2011. – С. 129-133 (см. http://immod.gpss.ru/)