

Опубликовано в материалах международной научной конференции «Компьютерные науки и информационные технологии» .– Саратов: СГУ, 2009.– с. 240–241.

КОМПОНЕНТНАЯ МОДЕЛЬ С ПОДДЕРЖКОЙ ДИНАМИЧЕСКОЙ РЕКОНФИГУРАЦИИ

П.Ю. Маврин, Г.А. Корнеев, А.С. Станкевич

Санкт-Петербургский государственный университет
информационных технологий, механики и оптики, СПб, Россия

В последнее время большой интерес вызывают *динамические компонентные системы*. В таких системах отдельные компоненты могут быть запущены, остановлены или изменены без перезапуска всей системы. Это удобно использовать в ответственных системах, которые нежелательно перезапускать целиком.

Наиболее известной моделью для построения динамических компонентных систем является модель OSGi [1]. Однако она обладает важным недостатком: она не гарантирует согласованность системы в процессе реконфигурации.

Авторами предлагается модель для построения динамических компонентных систем, лишенная указанного недостатка. В предлагаемой модели ядро передает компоненте ссылки на компоненты, предоставляющие нужные ей сервисы, при этом выполняется два следующих свойства.

Первое свойство: компонента, используемая работающей компонентой, также работает. Выполнение этого свойства позволяет компоненте в любое время совершать вызовы методов используемых сервисов, не опасаясь, что компонента, предоставляющая его, остановлена.

Второе свойство: ссылка, переданная ядром, всегда указывает на одну и ту же компоненту. Это свойство гарантирует, что серия последовательных вызовов методов используемого сервиса совершатся на одной и той же компоненте.

Ключевую роль в предлагаемой модели играет *декларативное объявление сервисов* [2], то есть каждая компонента заранее, до запуска, извещает ядро системы о том, какие сервисы она предоставляет и использует. Это позволяет ядру следить за существующими зависимостями между компонентами, запускать и останавливать их в нужном порядке. Отметим, что этот подход успешно используется в статических компонентных системах, например в ядре Spring [3]. Однако в таких системах задача поддержания согласованности намного проще, так как компоненты никогда не останавливаются.

Ядро системы в предлагаемой модели обрабатывает два вида событий: «компонента добавлена» и «компонента удалена». Для каждого события находятся все компоненты, которые оно затрагивает, после чего эти

компоненты запускаются или останавливаются в таком порядке, чтобы выполнялись свойства, описанные выше.

Проиллюстрируем работу ядра на простом примере. Пусть компонента *A* использует сервис *S*, предоставляемый компонентой *B*. Тогда если компоненту *A* добавят в систему до компоненты *B*, она не сможет быть запущена сразу. Однако после того как компонента *B* будет добавлена и запущена, ядро определит, что условия, необходимые для запуска *A*, выполнены и запустит ее. Аналогично, если компонента *B* будет удалена до компоненты *A*, то ядро сначала остановит *A*, а затем *B*.

Отметим, что предлагаемая модель не предполагает дополнительной работы для разработчика компонент, так как вся логика, связанная с запуском, остановкой и связыванием компонент реализована в ядре.

Модель реализована в ядре Taiga для построения динамических компонентных систем на языке Java.

Список литературы

1. *OSGi Alliance*. OSGi Service Platform Core Specification. 2007.
2. *Fowler M.* Inversion of Control Containers and the Dependency Injection pattern. <http://martinfowler.com/articles/injection.htm>
3. *Walls C., Breidenbach R.* Spring in Action (In Action series). Manning Publications Co. 2005.