Глава 7. Параллельность.

Мы понимаем параллельность мира

Глубокое понимание параллельности окружающего нас мира глубоко впечатано в наши мозги. Мы реагируем на внешние раздражители очень быстро при помощи отдела нашего мозга, называемого *амигдала.* Без такого реагирования мы бы довольно скоро погибли. Наше сознание, по сравнению с подобными реакциями, гораздо медленнее. За то время, пока сама мысль "нажми на тормоз" сформируется в нашей голове, это действие будет нами уже выполнено.

Когда мы ведем автомобиль в насыщенном транспортном потоке, мы ментально отслеживаем местоположение и скорость десятков, если не сотен автомобилей. Все это происходит без участия сознания. Если бы мы не умели это делать, мы бы, вероятно, погибли бы на дороге.

Наш мир параллелен

Если мы хотим писать программы, которые ведут себя подобно другим объектам реального мира, тогда эти программы должны быть параллельными по своей внутренней структуре.

Вот почему мы должны программировать на параллельном языке программирования.

А мы пока чаще программируем приложения для реального мира с помощью последовательных языков. Это излишнее усложнение.

Используйте язык, который был специально разработан для написания параллельных приложений и разработка параллельных систем станет намного проще и приятнее.

Программы на Эрланге моделируют наш процесс мышления и взаимодействия.

У нас с вами нет общей разделяемой памяти. У меня есть своя память, а у вас - своя. Каждый из нас имеет свои мозги, по одному на брата. Они не соединены вместе. Чтобы изменить вашу память, я должен послать вам сообщение: что-то сказать или, хотя бы, помахать рукой.

Вы слушаете, вы смотрите и ваша память изменяется. Но, тем не менее, не задав вам соответствующего вопроса или не видя вашей реакции, я не могу быть уверенным, что вы получили мое сообщение.

Чтобы убедиться в том что другой процесс получил ваше сообщение и изменил свою

память, вы должны спросить его об этом (послав ему сообщение). Именно так мы сами и взаимодействуем.

Сью: Привет, Билл, мой номер телефона - 45 67 89 12

Сью: Ты меня понял?

Билл: Конечно, твой номер телефона - 45 67 89 12.

Подобные паттерны взаимодействия очень хорошо нам известны. От самого рождения мы учимся, как взаимодействовать с окружающим миром наблюдая за ним, а также посылая разного рода сообщения и наблюдая за ответной реакцией.

Люди функционируют, как независимые сущности, которые общаются с помощью посылки друг другу сообщений.

Именно так работают процессы в Эрланге и именно так работаем и мы сами, а это значит, что нам будет очень просто понять механизмы работы программ написанных на Эрланге.

Программы на Эрланге состоят из десятков, сотен, а иногда и сотен тысяч маленьких процессов. И все они работают независимо друг от друга. Они общаются между собой путем посылки друг другу сообщений. Каждый процесс имеет свою собственную память. Они ведут себя как некое громадное сборище людей, которые общаются друг с другом.

Это позволяет значительно проще управлять и масштабировать программы на Эрланге. Предположим у нас есть десять людей (процессов) и слишком много работы, которую они должны сделать. Что мы можем сделать в такой ситуации? Позвать на помощь больше людей. А как нам управлять этими группами людей? Это просто - надо просто рассказать им, всем сразу, их инструкции (широковещательное сообщение от одного ко многим).

Процессы в Эрланге не имеют общей памяти, так что нет никакой необходимости в ее блокировке перед ее использованием. Там где есть замки (блокировки) там будут и потерянные ключи к этим замкам. А что происходит, когда вы теряете свой ключ? Вы паникуете и не знаете, что же вам теперь делать. И то же самое происходит в программных системах, теряются ключи и ваши блокировки заклинивает.

Распределенное программное обеспечение с блокировками и ключами к ним, всегда подвержено этому риску и страдает от этого.

А в эрланге просто нет никаких блокировок и ключей.

Если кто-то умирает, другие люди это замечают.

Если я, находясь в помещении с людьми, вдруг упаду и умру, то кто-то, вероятно, это заметит (по крайней мере, я на это надеюсь). Процессы Эрланга в этом также очень похожи на людей - они могут внезапно умереть. Но в отличии от людей, когда они умирают, они громко выкрикивают на своем последнем вздохе точную причину от чего они умерли.

Представьте себе комнату полную людей. Неожиданно кто-то из них падает и умирает. И в самый момент своей смерти он произносит: " Я умер от сердечного приступа" или "Я умер от разрыва слепого отростка желудка". Также поступают и процессы Эрланга. один процесс, умирая, может сказать "Я умер, потому что меня попросили разделить на ноль". А другой, возможно, скажет "Я умер, потому что меня спросили каков последний элемент в пустом списке".

А теперь, давайте представим, что в нашей комнате полной людей, есть специально назначенные люди, работа которых заключается в том, чтобы позаботиться о трупах. Давайте представим себе двух таких людей, Джейн и Джона. Если умирает Джейн, то Джон решает все проблемы, связанные с ее смертью. А если умирает Джон, то все эти проблемы решает Джейн. То есть, получается, что Джейн и Джон как бы связаны между собой незримым соглашением, в котором говорится, что если один из них умирает, то другой разбирается со всеми проблемами связанными с этой смертью.

Именно так работает механизм ошибок в Эрланге. Процессы в нем могут быть связаны вместе. И если один из них умирает, то другие получат сообщение об ошибке, говорящее о том что этот процесс умер.

Вот, в основном, и все.

Вот так и работают программы на Эрланге.

Итак повторим, то что мы уже узнали:

Эрланг программы состоят из множества процессов. Эти процессы посылают друг другу сообщения

Эти сообщения могут быть и не приняты адресатом и не поняты им. Если вы хотите точно узнать, что ваше сообщение было получено и понято, вы должны послать соответствующее сообщение этому процессу и ждать его ответа.

пары процессов могут быть связаны вместе. Если один из таких связанных процессов умирает, другому процессу из этой пары будет послано сообщение, содержащее в себе указание на причину смерти первого процесса.

Эту простую модель программирования я называю *параллельно ориентированное программирование.*

**

В следующей главе мы начнем писать параллельные программы. Нам потребуется для этого изучить три новых базовых механизма: порождение нового процесса (**spawn**), посылка сообщения (с помощью оператора **!**) и получение сообщения (**receive**). После чего мы сможем написать несколько простых параллельных программ.

Когда процесс умирает, некоторые другие процессы получают об этом сообщение, если они были с ним связаны. Это все рассматривается в Главе 9 *Ошибки в параллельных программах*.

Когда вы будете читать следующие две главы не забывайте о модели множества людей в одной комнате. Люди это процессы. Люди в комнате имеют каждый свою собственную память - это состояние процесса. Чтобы изменить вашу память, я должен вам что-то сообщить, а вы должны это воспринять и понять. Это и есть посылка и получение сообщений. У нас бывают дети - это порожденные нами процессы. Мы умираем - так заканчивается существование и работа процесса.