What if we can add new features without changing code?

Francis Leung

Can we add new functionality without changing code?

So What?

Can we verify software by assertions instead of case by case testing?

Outline

- On the problem of changing code
- What is FLX?
- An example: autonomic protocols one concern at a time
- Present projects

The problem of changing code

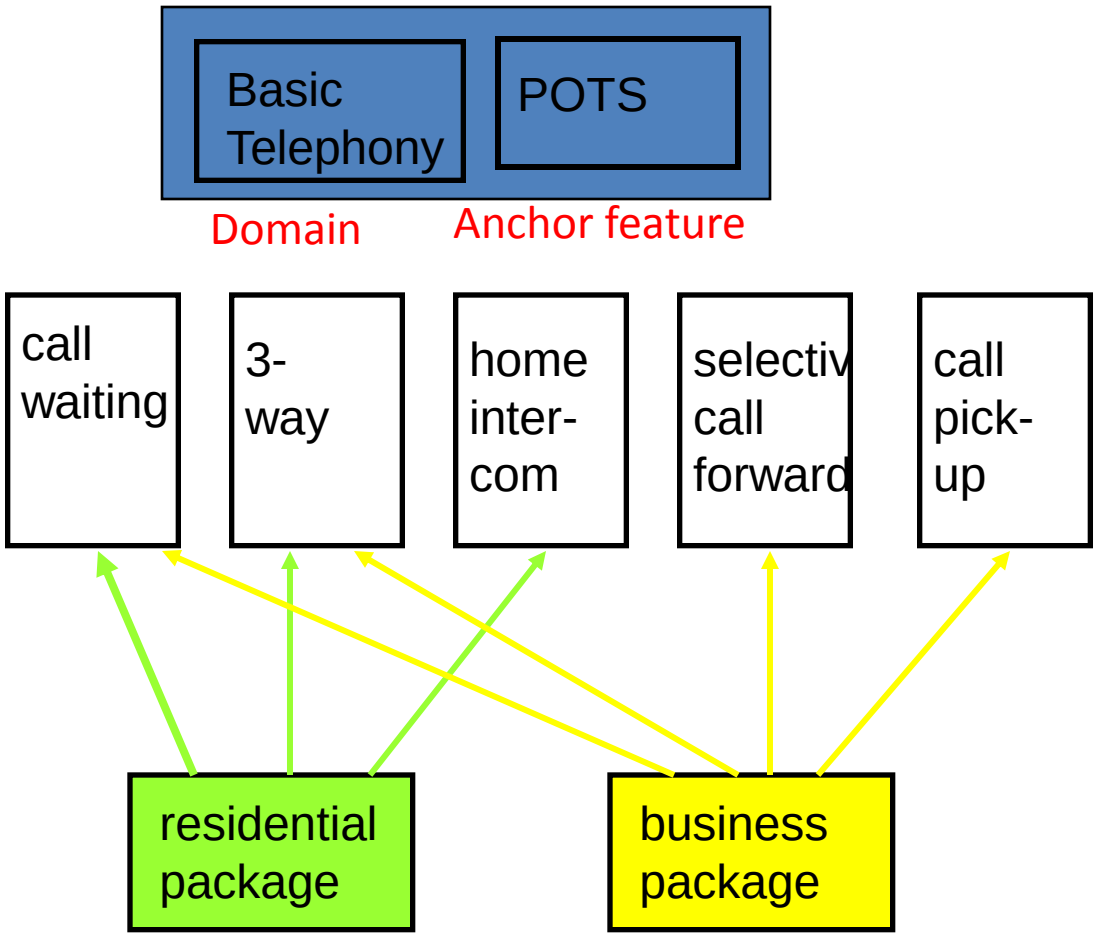
- Practically important: >90% of software development cost involves changing code
- Theoretically challenging: as challenging as the program proving problem
- Problem cannot be solved with existing mainstream programming languages when the features are interacting
  - Two features interact if one changes the behavior of the other when integrated together

The problem of changing code II

1. Programmer must look for where to change code
2. Programs of different concerns are entangled in the same reusable program module
3. The entangled concerns are not reusable without one another
4. Implementation of a concern is by changing the code of other concerns

What is FLX?

FLX Constructs



Model

Features

Features Packages

The challenges of the autonomic protocols

- A distributed system should be self healing, self optimizing, self configurating and self protecting
- These properties have many interacting concerns

Many concerns

Feature*	Concerns Covered	Concerns Raised
Neighbor detect	Detects an out of service	C1: What if the detector is faulty? C2: Scalability
Voting (2P)	Covers C1	C3.1: Blocking when coordinator out of service C3.2: State inconsistency when voter or coordinator out of service
Voting (3P)	Covers C3	C4: Coordinator single point of failure C5: State inconsistency when multiple out of services
Standby	Covers C4	C6: Need multiple standby's
Election	Covers C6	
Load reduction by grouping (dynamic)	Covers C2	C7: How to make group info reliable and resilient?
FLX feature packaging	Nodes may play different roles	
…	…	…

* Feature: a solution to one or more concerns

Potential of separation of concern

- Now the code for all these concerns must entangle

We want to:

- Simplify the problem by dealing with one concern at a time
- Allow application to “mix and match” the features that it wants

What are we working on?

- Design and integration methods
  - Software evolution
  - Comparative study
- Distributed systems
  - Object remoting in FLX
  - Autonomic protocols
- Tools
  - Design and verification