```
@startuml
!theme toy
skinparam linetype ortho
skinparam packageStyle rectangle

' --- CAPA DE DOMINIO (El Corazón - Sin dependencias externas) ---
package "Domain Layer" #f9f9f9 {

    package "Aggregates & Entities" {
        class User <<Aggregate Root>> {
            - UserId id
            - Email email
            - PasswordHash password
            - PlanType plan
            - Counter emailsSentThisMonth
            + boolean canSend(int amount)
            + void trackUsage(int amount)
        }

        class Campaign <<Aggregate Root>> {
            - CampaignId id
            - String subject
            - String contentHtml
            - SenderIdentity sender
            - CampaignStatus status
            - ListId targetListId
            + void launch()
            + void markAsFailed(String reason)
        }

        class Contact <<Entity>> {
            - ContactId id
            - Email email
            - Map<String, String> customFields
            + void updateCustomData(String key, String value)
        }
    }

    package "Value Objects" {
        class Email <<ValueObject>> {
            - String address
            + void validate()
        }
        class SenderIdentity <<ValueObject>> {
            - String name
            - String replyTo
            + String formatBranded(String domain)
        }
        enum PlanType {
            FREE(500)
            PRO(50000)
            - int limit
            + int getLimit()
        }
        enum CampaignStatus {
            DRAFT, SENDING, SENT, FAILED
        }
    }

    package "Ports (Interfaces)" {
        ' Puertos de Salida (Output Ports)
        interface UserRepository <<Port>>
        interface CampaignRepository <<Port>>
        interface ContactRepository <<Port>>
        interface EmailSender <<Port>>

        ' Puertos de Entrada (Input Ports - Opcional en algunas implementaciones, aquí son los UseCases)
    }
}

' --- CAPA DE APLICACIÓN (Orquestación de Casos de Uso) ---
package "Application Layer" #e3f2fd {

    class CreateCampaignUseCase {
        - CampaignRepository campaignRepo
        + CampaignId execute(CreateCampaignCommand cmd)
    }

    class ExecuteCampaignUseCase {
        - CampaignRepository campaignRepo
        - ContactRepository contactRepo
        - UserRepository userRepo
        - EmailSender emailSender
        + void execute(Long campaignId)
    }

    class ImportContactsUseCase {
        - ContactRepository contactRepo
        - UserRepository userRepo
        + void execute(ImportCommand cmd)
    }
}
```

```
' --- CAPA DE INFRAESTRUCTURA (Detalles de Implementación) ---
package "Infrastructure Layer" #fff3e0 {

   package "Persistence Adapters" {
      class JpaUserRepository implements UserRepository
      class JpaCampaignRepository implements CampaignRepository
      class JpaContactRepository implements ContactRepository
   }

   package "External Services Adapters" {
      class AwsSesAdapter implements EmailSender {
         - SesV2Client sesClient
      }
   }

   package "Security" {
      class JwtTokenProvider
      class AuthDetailsService
   }
}

' --- CAPA DE ENTRADA (Drivers / Controllers) ---
package "Web Layer (Adapters In)" #f1f8e9 {
   class CampaignController {
      - CreateCampaignUseCase createUseCase
      - ExecuteCampaignUseCase executeUseCase
      + create()
      + send()
   }
   class ContactController {
      - ImportContactsUseCase importUseCase
      + batchUpload()
   }
}

' Relaciones principales
CampaignController --> CreateCampaignUseCase
CampaignController --> ExecuteCampaignUseCase
ExecuteCampaignUseCase --> EmailSender : "uses Port"
ExecuteCampaignUseCase --> CampaignRepository : "uses Port"
ExecuteCampaignUseCase --> ContactRepository : "uses Port"
ExecuteCampaignUseCase --> UserRepository : "uses Port"

' El Dominio no conoce a nadie, la Infraestructura implementa sus Puertos
JpaCampaignRepository ..|> CampaignRepository
AwsSesAdapter ..|> EmailSender
```

```
@enduml
```