گزارش فاز چهارم پروژه تحلیل و طراحی سیستم‌ها

دانشور امراللهی ۸۱۰۱۹۷۶۸۵
علیرضا آقایی ۸۱۰۱۹۷۶۷۹
مهیار کریمی ۸۱۰۱۹۷۶۹۰

نمودار کلاس برای مورد کاربرد درخواست انبارداری:

روابط:

- هر System شامل تعدادی Customer و Storage می‌باشد. بنابراین از کلاس System به دو کلاس ذکر شده، یال Association داریم
- کلاس Customer متدهایی با پارامترهایی از جنس کلاس‌های Payment و Storage دارند. بنابراین از کلاس Customer به کلاس‌های ذکرشده یال Dependency داریم.
- هر Storage شامل دقیقا یک Storage Supervisor است. بنابراین از کلاس Storage به StorageSupervisor یال Association داریم.

متدها:

- برای هر Customer به کمک متد sendRequest نوع درخواست و خواستن/نخواستن کارشناس مشخص می‌شود.
- برای هر Customer به کمک متد payRequest پرداخت انجام می‌شود.
- برای هر Customer به کمک متد deposit مابه التفاوت پرداخت می‌شود.

- هر System به کمک متد specifyArea از مشتری تخمین فضای بار را دریافت می‌کند.
- هر System به کمک متد checkForArea از انبار، وجود فضای درخواستی را بررسی می‌کند.
- هر System به کمک متد selectTimeSlot به مشتری ساعت‌های بازدید را ارائه می‌دهد.

- هر StorageSupervisor به کمک متد reportOnDeposit گزارش پرداخت انتهایی را به System ارائه می‌دهد.

- برای هر Storage، به کمک متد handleDeposit مقدار deposit بر اساس حجم بار و مابه‌التفاوت ثبت می‌شود.

در ادامه تطابق این طراحی را با اصولی از GRASP که در این طراحی قابل بررسی هستند، مشخص می‌کنیم:

- می‌دانیم در طراحی ما، System بعنوان Container برای Requestها عمل می‌کند. بنابراین، متد SendRequest که برای Customer در نظر گرفته شده است، در نهایت به ساخته شدن یک شیء از نوع Request در System خواهد شد و اصل Creator حفظ می‌شود. برای موجودات از نوع Payment نیز ابتدا توسط System ویژگی link آن‌ها ساخته می‌شود و سپس خود Payment توسط Customer و با کمک link که از System به او پاس داده می‌شود ساخته می‌شود.
- در این طراحی سعی شده است که توابعی مانند checkForArea که در شرایط واقعی نیز بایستی توسط Storage انجام شوند، متدی از همین کلاس در نظر گرفته شوند. بعنوان مثالی دیگر، تابع reportOnDeposit به‌عنوان متدی از StorageSupervisor در نظر گرفته شده است. به این ترتیب، اصل فاعل متخصص نیز حفظ می‌شود.
- در این طراحی سعی شده است که متدهای هر کلاس متناسب با هدف آن کلاس باشند تا بیشترین Cohesion در این طراحی صورت بگیرد. اصل Low Coupling نیز به‌عنوان نتیجه‌ای از High Cohesion رعایت شده است.

نمودار کلاس برای مورد کاربرد درخواست جابه‌جایی:

روابط:

- هر System شامل تعدادی Evaluator، Customer و MovingTeam می‌باشد. بنابراین از کلاس System به سه کلاس ذکر شده، یال Association داریم
- کلاس Customer متدی از جنس کلاس‌های Payment دارند. بنابراین از کلاس Customer به کلاس ذکرشده یال Dependency دارد.
- هر MovingTeam شامل دقیقا یک Moving Supervisor است. بنابراین از کلاس MovingTeam به Moving Supervisor یال Association داریم.

متدها:

- برای هر Customer به کمک متد sendRequest نوع درخواست و خواستن/نخواستن کارشناس مشخص می‌شود.
- برای هر Customer به کمک متد payRequest پرداخت انجام می‌شود.

- هر System به کمک متد selectTimeSlot به مشتری ساعت‌های بازدید را ارائه می‌دهد.
- هر System به کمک متد Pay به مشتری لینک پرداخت را ارائه می‌دهد.
- هر System به کمک متد Evaluate به کارشناس دستور کارشناسی می‌دهد.
- هر System به کمک متد Schedule برنامه جابه‌جایی را به MovingTeam می‌دهد.

- هر StorageSupervisor به کمک متد reportOnDeposit گزارش پرداخت انتهایی را به System ارائه می‌دهد.

- برای هر MovingTeam به کمک متد Status از MovingSupervisor استعلام وضعیت می‌کند.
- برای هر MovingTeam به کمک متد PayRemaining از Customer درخواست پرداخت مابقی پول می‌شود (نقدی و نه در سیستم)

- برای هر MovingTeam به کمک متد HandleRemaining گزارش پول نقدی دریافت‌شده را به سیستم می‌دهد.