به نام خدا

دانشگاه اصفهان دانشکده مهندسی کامپیوتر

تمرین اول

افزونه یدکی

نام درس	معماری نرم افزار
استاد درس	دکتر محمدرضا شعرباف
طراح	محمد خورسندی

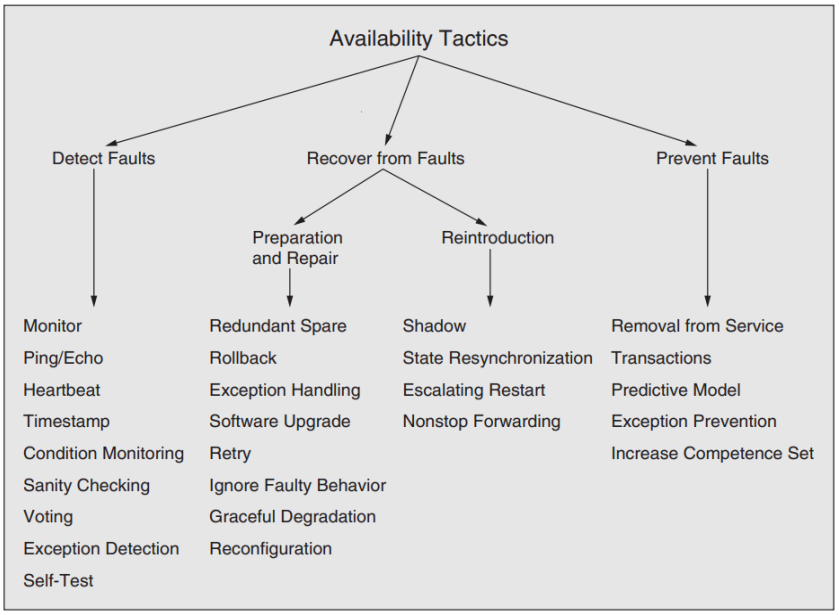
تابستان ۱۴۰۴

فهرست مطالب

مقدمه

دسترس‌پذیری[1] یکی از ویژگی‌های کلیدی در معماری نرم‌افزار است که بیان می‌کند یک سیستم تا چه حد در زمان نیاز، در دسترس و قابل استفاده است. در بسیاری از سیستم‌های حیاتی مانند خدمات بانکی، فروشگاه‌های آنلاین، سامانه‌های اورژانس و ...، حتی چند ثانیه قطعی می‌تواند خسارت‌های مالی یا اعتباری جدی به همراه داشته باشد. برای افزایش دسترس‌پذیری، سیستم باید بتواند در برابر خرابی‌ها مقاومت کند یا به‌سرعت بازیابی شود.

به طور کلی سه دسته تاکتیک برای افزایش دسترس‌پذیری نرم افزار وجود دارد: جلوگیری از خطا، شناسایی خطا و بازیابی از خطا. در شکل زیر می‌توانید این سه دسته را مشاهده کنید:



یکی از تاکتیک‌هایی که برای بازیابی از خطا استفاده می‌شود، **افزونه یدکی**[2] است که در صورت خرابی مؤلفه‌ی اصلی، جایگزین آن می‌شود و عملکرد سیستم را حفظ کنند. سه الگو معماری زیر بر این تاکتیک بنا شده‌اند:

- **یدک داغ**[3] مؤلفه‌ی پشتیبان به‌صورت همزمان با مؤلفه‌ی اصلی در حال اجرا و به‌روز است و در صورت خرابی، بلافاصله می‌تواند جایگزین شود.
- **یدک گرم**[4] مؤلفه‌ی پشتیبان فعال است اما داده‌های آن ممکن است کمی قدیمی باشد؛ برای جایگزینی نیاز به همگام‌سازی سریع دارد.

---

[1] Availability

[2] Redundant Spare

[3] Hot Spare

[4] Warm Spare

- **یدک سرد**[5] مؤلفه‌ی پشتیبان غیرفعال است و تنها در زمان خرابی مؤلفه‌ی اصلی راه‌اندازی می‌شود و ممکن است زمان‌بر باشد.

در این تمرین، هدف شما پیاده‌سازی و تحلیل این سه الگوهای به‌عنوان راهکارهایی برای افزایش دسترس‌پذیری سیستم است.

۱- ساختار اولیه پروژه

پروژه اولیه در این لینک موجود است. در این پروژه، یک سیستم ساده شامل مولفه‌های زیر طراحی شده است:

- دو سرور با نام‌های server و spare که هر دو یکسان و پیاده سازی یک سیستم ذخیره و بازیابی کتاب هستند.
- یک **پروکسی**[6] که درخواست‌های کاربر را دریافت می‌کند و به یکی از سرورها می‌فرستد. به علاوه این پروکسی قطعی سرورها را تشخیص می‌دهد و در صورت آماده بودن سرور دیگر به آن سوئیچ می‌کند.
- رابط کاربری شامل دو صفحه است.
- صفحه‌ی ایجاد، حذف و مشاهده کتاب‌ها (/)
  - با استفاده از این صفحه می‌توان بررسی کرد داده‌ها بعد از تغییر سرور حفظ شده و به درستی بازیابی شوند.
- صفحه‌ی مانیتورینگ سرور برای شبیه‌سازی خطا (/monitor)
  - با استفاده از این صفحه می‌توان یک خطا در سیستم شبیه‌سازی کرد به طوری که سرور از دسترس خارج شود.
  - بعد از کلیک، یک تایمر شروع می‌شود و وقتی یک سرور جدید بالا آمد، پایان می‌یابد. از این تایمر برای تحلیل دسترسی پذیری استفاده کنید.

شما باید روی همین پروژه، سه الگوی ذکر شده را پیاده‌سازی کنید.

۲- یدک داغ

- در این حالت، هر دو سرور هم‌زمان و کاملاً فعال هستند.
- پروکسی باید در لحظه‌ای که یکی از سرورها از کار می‌افتد، به سرور دیگر سوییچ کند.
- در این مدل، هر تغییر (مثلاً افزودن/حذف کتاب) باید هم‌زمان روی هر دو سرور اعمال شود.

---

[5] Cold Spare

[6] proxy

۳- یدک گرم

- در این مدل، تنها سرور اصلی فعال است.
- سرور یدک روشن و آماده است ولی مستقیماً سرویس نمی‌دهد.
- داده‌ها در بازه‌های زمانی مشخص با سرور اصلی همگام‌سازی می‌شوند (ترجیحاً با استفاده از API)
- وقتی سرور اصلی از کار می‌افتد، پراکسی باید به سرور پشتیبان سوئیچ کند.

۴- یدک سرد

- فقط سرور اصلی فعال است.
- زمانی که سرور اصلی خراب شود، سرور پشتیبان **به صورت خودکار روشن** می‌شود.
- برای بازیابی داده‌ها می‌توانید از ذخیره لاگ‌ها و یا پایگاه داده سرور اصلی استفاده کنید.
- پروکسی باید متوجه شود که سرور اصلی خاموش شده است، و وقتی سرور پشتیبان آماده شد، به آن سوییچ کند.

۵- نکات مهم برای پیاده‌سازی

- از ping که در سرورها پیاده‌سازی شده می‌توانید برای بررسی سلامت سرورها استفاده کنید. هر چند مکانیزم بررسی سرورها در پروکسی پیاده‌سازی شده، در صورت نیاز می‌توانید این مکانیزم را در جهت بهبود کارایی تغییر دهید.
- بهتر است برای پروکسی از پورت ۸۰۰۰، برای سرور اصلی ۸۰۰۱ و برای سرور یدکی از ۸۰۰۲ استفاده کنید.
- نحوه راه اندازی پروژه در فایل README در مخزن داده شده قرار داده شده است.
- توجه کنید که همه درخواست‌ها باید به پروکسی ارسال شوند و رابط کاربری نباید مستقیم به سرورها درخواست ارسال کند.

۶- آنچه باید تحویل داده شود

- برای هر کدام از الگوهای بالا یک برنچ در گیت (مجموعا سه برنچ) بسازید و پیاده سازی مربوط به هر الگو را در برنچ مربوطه انجام دهید.
- یک گزارش مختصر از پیاده‌سازی هر یک از الگوها ارائه دهید. گزارش بایستی شامل تصمیمات کلیدی و تاثیر گذار بر کارایی مانند نحوه همگام سازی داده‌ها و جایگزینی سرور باشد. از ذکر جزئیات در گزارش پرهیز کنید.
- با استفاده از صفحه پایش (monitor) موجود در رابط کاربری میانگین زمان بازیابی هر حالت را محاسبه و سپس دسترسی پذیری هر حالت را با استفاده از فرمول زیر محاسبه کنید. (فرض کنید سیستم به طور میانگین یک بار در روز خرابی دارد)

$$\boldsymbol{MTBF/(MTBF + MTTR)}$$

where *MTBF* refers to the mean time between failures and *MTTR* refers to the mean time to repair. In the software world, this formula should be interpreted to mean that when thinking about availability, you should think about what will make your system fail, how likely it is that such an event will occur, and how much time will be required to repair it.