DT024G - Projektbeskrivning

Lennart Franked

6 maj 2026

Innehåll

1 Introduktion

Syftet med detta projekt är att ge dig praktisk erfarenhet av hur protokoll på applikationsnivå kan utformas och implementeras.

Genom att implementera en enkel klient-server-applikation kommer du att få en praktisk förståelse för hur processer kommunicerar över ett nätverk med hjälp av sockets.

Du uppmuntras att hålla applikationen enkel och fokusera på korrekthet, tydlighet och robusthet snarare än komplexitet.

Applikationen ska bestå av två separata program: en server och en klient. Servern ska lyssna efter inkommande anslutningar, medan klienten initierar kommunikationen. Kommunikationen kan vara textbaserad och ska demonstrera en tydlig interaktion av typen begäran-svar.

Exempel på möjliga applikationer inkluderar (men är inte begränsade till):

- En enkel chattapplikation.
- Ett quiz- eller gissningsspel.
- En kalkylatortjänst.
- En enkel informations- eller tidsserver.

Förutom att skriva en enkel server-klient-applikation ska du även skriva ett teoriavsnitt som behandlar applikationslagret.

Detta är i första hand en nätverksuppgift, inte en programmeringsuppgift. Det är därför tillåtet att använda språkmodeller *enbart som stöd* vid utveckling av koden.

Språkmodeller får *inte* användas för att producera rapporttext. När rapporten är färdigställd är det däremot tillåtet att låta en språkmodell granska den, till exempel för återkoppling på språk, struktur och tydlighet. Eventuella ändringar ska du själv ta ställning till och formulera.

Målet är att förstå kommunikationsprinciper, inte att bygga ett komplext system.

1.1 Grupparbete

Du får genomföra projektet ensam eller i grupp om två. Om ni arbetar i par förväntas omfattningen reflektera detta:

- Två protokoll på applikationslagret ska förklaras ingående i teoridelen (istället för ett).
- Programmet förväntas vara större och mer ambitiöst än ett enmansprojekt.

2 Mål

Denna projektuppgift syftar till att hjälpa dig att uppnå följande av kursens lärandemål:

- kategorisera olika datakommunikationsprotokoll i rätt lager i TCP/IP- och ISO OSI-modellen.
  - Genom att skriva ett avsnitt som behandlar applikationslagret.
  - Genom att implementera och analysera ditt eget protokoll får du praktisk insikt i hur de teoretiska protokollagren förhåller sig till verklig kommunikation.
- förstå och förklara funktionen hos olika nätverkskomponenter
  - Genom att skapa en server-klient-applikation.
- beskriva hur olika adressarkitekturer förhåller sig till varandra.
  - Genom att sätta upp nätverkssockets får du en djupare förståelse för hur portnummer och IP-adresser hänger ihop.
  - Genom att inspektera de fångade paketen i Wireshark får du även se hur MAC-adresser i Ethernet-ramen relaterar till IP-adresser och portnummer.
- använda olika verktyg för felsökning av nätverk.
  - Genom att använda nätverksanalysatorn Wireshark för att fånga och inspektera data som skickas mellan din server och klient.

3 Läsanvisningar och förberedelser

Innan du påbörjar projektet förväntas du studera följande material för att förstå den underliggande teorin och de praktiska tekniker som krävs för att genomföra uppgiften.

En del av denna inläsningsuppgift inkluderar:

- *Data communications and networking with TCP/IP protocol suite*, Kapitel 10[1]
- *Socket Programming HOWTO*[2]

Till din hjälp finns ett mycket enkelt ekoprogram tillgängligt på kursplattformen. Programmet är skrivet i Python och kommunicerar över TCP.

Du får återanvända strukturella delar av exempelkoden, som stöd för hur man utvecklar med socketprogrammering. Men du måste utforma ditt eget applikationsprotokoll.

4 Projektkrav

Din uppgift i detta projekt är att skriva en egen nätverksapplikation i Python (men du är fri att använda ett annat språk om du vill). Följande kriterier måste

uppfyllas:

- Det ska vara ett enkelt klient-server-program.
- Du ska ha utformat ett enkelt protokoll på applikationslagret som inkluderar följande delar.
  - Upprätta en anslutning (Handskakning)
  - Utbyte av applikationsdata
  - Stänga anslutningen (Avslutning)
- Protokollet måste tydligt definiera ordningen och betydelsen av de meddelanden som utbyts mellan klient och server.
- Klienten och servern måste kommunicera med hjälp av sockets.
- Du får välja det transportlagerprotokoll som bäst passar din applikation.
- Vad din nätverksapplikation ska göra är upp till dig att bestämma.
- När din applikation fungerar ska du med hjälp av Wireshark [3] fånga datautbytet. Inspelningen kommer att användas för att analysera hur ditt applikationslagerprotokoll transporteras över transportlagret.

Återigen ligger fokus på att visa förståelse för nätverkskoncept snarare än att bygga ett stort system.

5 Disposition och krav på projektrapporten

Nedan finner ni angivet vad er projektrapport måste innehålla.

5.1 Introduktionsdel

Introduktionskapitlet måste innehålla följande:

- Bakgrundskapitel, vad handlar projektet om.
- Syfte, Vad är målet med just ditt projekt.
- Mål. Definiera tre konkreta mål kopplat till just ditt projekt som du kan använda för att verifiera att ditt projekt uppfyller kraven för uppgiften. Om du och en klasskamrat kan använda samma mål rakt av, är de för generellt skrivna.
- Författarnas bidrag (endast vid grupparbete om två). Beskriv tydligt hur arbetet har fördelats mellan författarna, t.ex. vem som ansvarat för protokolldesign, klient, server, testning, Wireshark-analys och rapportskrivning. Båda författarna ska ha bidragit aktivt till samtliga delar av projektet.

5.2 Teoridel

Teoriavsnittet måste innehålla följande:

- Vad är syftet med applikationslagret.

- Beskriv kort tre protokoll på applikationslagret, välj därefter ett av dem och skriv en mer ingående förklaring om hur det fungerar. (Arbetar ni i par ska två protokoll förklaras ingående.)
- Hur förhåller sig applikationslagret till transportlagret? I form av typ av data som skickas och funktion.
- Var kommer den data ifrån som placeras i applikationslagrets datafält? Beskriv vad som händer "ovanför" applikationslagret, det vill säga hur applikationen själv producerar och konsumerar data.
- En beskrivning om hur ert applikationslagerprotokoll ska fungera. Använd sekvensdiagram som stöd.

5.3 Metoddel

Metodavsnittet måste innehålla följande:

- Beskriv kortfattat hur du ska gå tillväga för att lösa denna uppgift.
- Hur kommer du att utvärdera din lösning för att vara säker på att de mål du listade i introduktionen är uppfyllda?
- Hur kommer du att designa dessa tester?
- Användning av AI. Redogör ingående för hur AI-verktyg har använts i arbetet. Beskriv vilka verktyg och modeller som använts, i vilka delar av projektet de använts (t.ex. idégenerering, design, implementation, felsökning eller skrivande), exempel på prompts som ställts, hur svaren bearbetats och verifierats, samt eventuella begränsningar du stött på. Syftet är att tydliggöra vad som är ditt eget arbete och hur AI har bidragit som stöd i processen.

5.4 Implementationsdel

Implementationskapitlet måste innehålla följande:

- Beskriv från en hög nivå hur ditt applikationslagerprotokoll fungerar.
- Till din hjälp använder du pseudokod eller sekvensdiagram (dia, visio).
- Plocka ut *relevanta* kodsnuttar från ditt program och beskriv dessa i detalj.

5.5 Resultatdel

Resultatkapitlet måste innehålla följande:

- Resultatet av den utvärdering du beskrev i metodkapitlet.
- Resultatet av de tester du beskrev i metodkapitlet.

5.6 Diskussionsdel

Diskussionskapitlet måste innehålla följande:

- En kortare självreflektion över ditt projekt och resultaten. Finns det några begränsningar i ditt protokoll?

- En slutsats där du tydligt visar att de mål du satt är uppfyllda, och att du därmed uppfyllt alla krav som denna projektuppgift har.

6 Examination

Detta projekt examineras genom en projektrapport, med tillhörande bilagor, samt en muntlig presentation.

6.1 Skriftlig rapport med bilagor

Den skriftliga rapporten måste uppfylla de krav som är angivet i avsnitt 5.

Förutom rapporten ska du även lämna in din källkod och din Wireshark-fil som visar ditt protokoll.

Den inlämnade koden ska vara körbar och tydligt strukturerad och kommenterad.

Deadline samt instruktioner för inlämning hittar du på kursplattformen.

6.2 Muntlig presentation

De avslutande kraven gäller din muntliga presentation:

- Tid: 5 min
- Innehåll:
  - Kort bakgrund om ditt protokoll och dina angivna projektmål (1min)
  - Presentation av din pseudokod/sekvensdiagram (2min)
  - Redovisning av dina resultat (1min)
  - Slutsats (1min)
- Efter presentationen ska du vara beredd på att besvara en eller två frågor från examinator.

6.3 Betygssättning

Projektuppgiften är U/G.

A Förslag på tidsbudget

Projektet är dimensionerat för cirka 40 timmars arbete (två veckor på halvfart, 20h/vecka). Tabell 1 ger ett förslag på hur du kan fördela tiden.

Referenser

[1] Behrouz A. Forouzan. *Data communications and networking with TCP/IP protocol suite*. Sixth edition. New York: McGraw Hill. ISBN: 9781260597820.

Moment	Tid
Inläsning (Forouzan kap. 10 + Python sockets)	4–6 h
Design av protokoll och formulering av projektmål	3–4 h
Implementation av server och klient	8–12 h
Wireshark-inspelning och analys	2–3 h
Rapportskrivning	12–16 h
Förberedelse av muntlig presentation	2–3 h
Totalt	ca 40 h

Tabell 1: Förslag på tidsfördelning för projektet.

[2] Gordon McMillan. *Socket Programming HOWTO*. `https://docs.python.org/3/howto/sockets.html`. Fetched: 2026-02-10.

[3] Wireshark Foundation. *Wireshark*. `https://www.wireshark.org`. Hämtad: 2026-05-06.