Love Johansson, Johan Larsson

Ett ramverk för informationsextrahering

Inom storskalig mjukvaruutveckling finns ett behov av en kontrollerad arkitektur för att systemet man utvecklar ska kunna gå att hantera, och inte minst vidareutveckla. Utan fastslagna riktlinjer och bestämmelser för arkitekturen, spenderas mer tid på att förstå systemet och underhålla det än att vidareutveckla det, och slutligen kommer arkitekturen i stort sett att bero på de enskilda utvecklarna. Därför har uteslutande alla stora företag idag någon form av mjukvaruarkitektur och riktlinjer och bestämmelser för denna. Problemet ligger i att denna arkitektur kan vara svår att verifiera på den slutliga produkten, då systemet i många fall är för stort för att kunna överblicka. Därför behövs ett enkelt sätt att utvinna information från systemets källkod, som sedan skulle kunna användas till att bland annat verifiera arkitekturen. För att kunna göra detta, måste man först komma till problemets kärna, vilket inte alltid är det enklaste. Hur vet man vilken information som är viktig? Är det överhuvudtaget möjligt att utvinna den viktiga informationen? Hur lagrar man den utvunna informationen? Hur använder man sedan informationen för att lösa problem?

Under ett halvår har vi utvecklat ett ramverk för detta ändamål. Detta har genomförts på Ericsson AB i Lund, och är därför speciellt anpassad för den källkod som förekommer i den plattform för mobila enheter som företaget tillverkar. Mjukvaran till denna är uteslutande skriven i programmeringsspråket C, och innehåller komplicerade lösningar för dels realtidskommunikation och dels för den varianthantering som krävs för att samma mjukvara ska kunna återanvändas med enbart små ändringar. Verktyget som vi utvecklat utifrån dessa förutsättningar är indelat i tre skilda system. Ett av systemen inriktar sig på att bearbeta källkoden, för att kunna lagra information i en databas. Detta system är uppbyggt så att koden först bearbetas av ett lexikaliskt analysverktyg, som delar upp koden i små beståndsdelar, och sedan sparar undan det berabetade formatet. Efter detta steg genomsöks dessa beståndsdelar av små enheter vars enda uppgift är att leta upp vissa språkliga konstruktioner, som t.ex variabeltilldelningar eller funktionsdefinitioner, och spara information om dessa. Dessa enheter är skrivna med hjälp av en egenutvecklad metod för att definiera funktioner som hittar mönster som motsvarar sådana konstruktioner. Eftersom de arbetar på ett mellanformat, behöver den lexikaliska analysen enbart genomföras en gång för hela systemet, varpå flera enheter kan tillämpas på resultatet hur många gånger som helst. Efter detta sparas de hittade språkkonstruktionerna i en relationsdatabas.

Den databaslösning vi har använt är baserad på Prolog, men systemet är skrivet så att man enkelt kan byta ut detta. Som slutanvändarprogram har vi konstruerat ett skelettsystem som innehåller funktionalitet för databasåtkomst till vårt Prologsystem, och till detta skelettsystem skriver sedan användaren tilläggsprogram. Dessa tilläggsprogram kan användaren sedan skräddarsy för vilken sorts uppgifter som helst, som t.ex att överföra information från databasen till ett presentationsprogram, eller visa informationen i en tabell, eller som en HTML-sida. Detta gör att vårt verktyg kan utföra kraftfulla Prolog-operationer på databasen, samtidigt som man kan visa resultatet på det format man önskar.

Handledare: **Lars Bendix** (LTH), **Göran Fries** (NatFak), **Le Chi Thu** (Ericsson)
Examensarbete 20p, HT2005
Institutionen för Datavetenskap