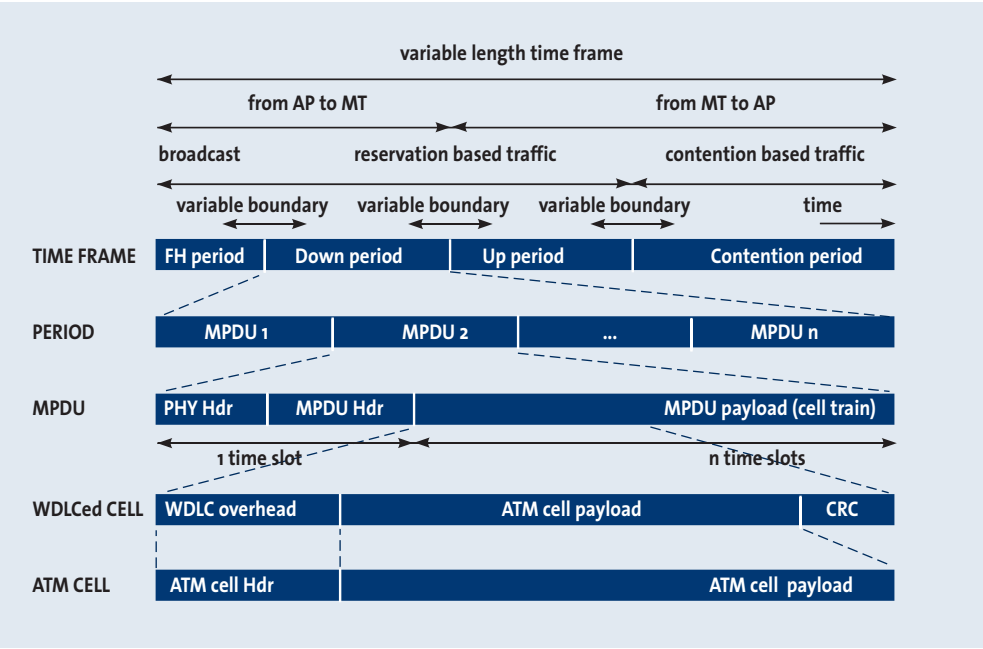
Nieuwe technieken leiden tot een fundamenteel andere

Model checking communicatieprotocol

Naarmate embedded systemen complexer van structuur worden, schieten traditionele simulaties tekort in het blootleggen van fouten in een ontwerp. Model checking is een aanvullende methode die relatief weinig extra expertise vereist omdat het in zekere zin 'simulatie in een hogere versnelling' is.

DR.IR. DENNIS DAMS Elektrotechniek en Informatica, Technische Universiteit Eindhoven.



1. Mascara timeframe. Data wordt verzonden in afzonderlijke tijdsegmenten (time frames), waarbij elk segment begint met een header waarin een 'inhoudsopgave' van de rest van het segment wordt gegeven.

Lange, monotone gangen, brede deuren: het decor van een ziekenhuis. Doktoren snellen rond met een notebook-computer onder hun arm geklemd. Bij de patiënt slaan zij het scherm open, voeren enkele gegevens in, en het beeld vult zich met actuele gegevens over de toestand van de patiënt. Gedigitaliseerde röntgenfoto's verschijnen met een simpele druk op de knop. Eventueel kan een directe voiceverbinding worden gemaakt met een medewerker van het laboratorium voor de uitslag van de laatste bloedtest.
Dit scenario is nog geen alledaagse praktijk, hoewel alle ingrediënten in principe voorhanden zijn. De vraag is alleen hoe deze te combineren. Het geschetste plaatje is één van de zogenoemde *usertrials* in het Europese Wand-project, wat staat voor 'Wireless ATM network demonstrator'. In dit project werkt de industrie samen met een aantal universiteiten aan de integratie van technieken die dit soort toepassingen dichterbij moeten brengen. Eén van de doelstellingen van het project is het daadwerkelijk realiseren van een demonstratie-opstelling binnen een ziekenhuis.

Een onderdeel van de binnen Wand ontwikkelde technieken, namelijk een communicatie-protocol genaamd Mascara, vormt het onderwerp van studie binnen een ander project op Europees niveau: Vires (zie kader). In Vires wordt het Mascara-protocol doorgelicht op diverse aspecten van correctheid. De belangrijkste methode die hierbij wordt gebruikt is *model checking*. Het navolgende geeft een korte introductie in deze methode, waarna de toepassing ervan in de context van Mascara wordt geschetst.

Model checking

Protocollen als Mascara zijn berucht om hun complexiteit; het is schier onmogelijk al tijdens het ontwerp alle mogelijke incorrecte scenario's te voorzien. Echter, hoe later een fout wordt gevonden, des te hoger zijn de kosten om deze te herstellen. Vanuit de wereld van systeemontwerp is er daarom grote belangstelling naar verbeterde technieken waarmee fouten in een vroeg stadium van het ontwerp kunnen worden onderschept. Een belangrijke randvoorwaarde daarbij is dat zo'n techniek kan worden ingepast in het bestaande ontwerptraject.
In het *debuggen* (traceren van fouten) van systemen neemt simulatie traditioneel een belangrijke plaats in. Een systeem wordt gemodelleerd, over het algemeen in software, en vervolgens worden diverse scenario's uitgetest op het model. Een dergelijk scenario kan

toetst correctheid

Mascara

Mascara staat voor 'mobile access scheme based on contention and reservation for ATM'. Dit communicatie-protocol moet zorgen voor een naadloze koppeling tussen ATM-technologie enerzijds en een draadloos medium anderzijds. Dat is een ambitieuze doelstelling. ATM (asynchronous transfer mode) is het antwoord van de telefoonindustrie op de vraag naar hogere datasnelheden in combinatie met hybride vormen van dataverkeer, bijvoorbeeld tekst gecombineerd met plaatjes en realtime audio. Het betekent een radicale breuk met de honderd jaar oude traditie van de zogenoemde circuitschakeling in het telefoonsysteem, waarbij er een fysieke (koperdraad-)verbinding tot stand wordt gebracht tussen twee partijen.
ATM daarentegen maakt gebruik van celschakeltechnologie. Hierbij wordt alle te verzenden data opgedeeld in cellen, kleine pakketten van een vaste grootte. Deze cellen worden dan afzonderlijk over het netwerk gestuurd. Een dergelijk idee ligt ook ten grondslag aan het bekende Internet Protocol (IP). Deze techniek biedt een grotere flexibiliteit in digitale verwerking van data. Een ander aspect van ATM is dat er tussen klant en aanbieder van het netwerk (carrier) een contract wordt afgesloten waarin de kwaliteit van de verleende dienst wordt vastgelegd.
Bij een openbaar netwerk heeft zo'n Quality of Service (QoS) contract een wettelijke status. Er kan bijvoorbeeld in gespecificeerd zijn hoe groot de maximale kans op verlies of verminking van cellen mag zijn voor verschillende types van dataverkeer, zoals email, spraak, en videobeelden. Om dergelijke garanties te kunnen bieden, vereist ATM over het algemeen dan ook hoge datasnelheden (vele megabits per seconde) en lage foutenkansen. De onderliggende fysieke basis is daarom idealiter een vast, *wired* netwerk.
Draadloze verbindingen daarentegen worden gekenmerkt door lage datasnelheden en hoge foutkansen. Daarnaast heeft radiotransmissie inherent een *broadcast*-karakter: uitgezonden signalen kunnen door willekeurig veel ontvangers tegelijk worden opgevangen. Deze karakteristieken staan lijnrecht tegenover de vereisten voor implementatie van ATM-technologie en hierin ligt dan ook de uitdaging voor een protocol als Mascara.
Zuinig omspringen met de beschikbare bandbreedte is essentieel, vandaar dat Mascara gebruik maakt van een schema van 'contention and reservation'. Data wordt verzonden in afzonderlijke tijdsegmenten (time frames, zie figuur 1), waarbij elk segment begint met een header waarin een 'inhoudsopgave' van de rest van het segment wordt gegeven. Zo weet iedere ontvangende partij, na het interpreteren van de header, welke delen van de rest van het segment voor hemzelf bedoeld zijn. Zo'n deel bevat over het algemeen meerdere ATM-cellen. Dit is het aspect van de 'reservation'. Omdat de inhoudsopgave per segment kan variëren, laat dit een zeer flexibele verdeling van de capaciteit toe. Daarnaast is er een stukje van elk tijdsegment vrijgehouden waarin nieuwe partijen zich kunnen aanmelden (contention), bijvoorbeeld als een mobiele terminal zojuist is aangeschakeld en zich wil aanmelden bij een access point.
De specificatie van het Mascara-protocol, inclusief de daarin veronderstelde eisen van correctheid, behelzen zeer diverse aspecten. Het concept van Quality of Service bijvoorbeeld stelt hoge eisen aan de manier waarop congestie van het (draadloze) medium wordt voorkomen en routering en scheduling van ATM-cellen wordt geregeld. Mobiele terminals moeten kunnen omschakelen tussen access points (handover) en wel op een zodanige wijze dat de verschillende ATM-kanalen die een mobiele terminal tegelijk in gebruik heeft, worden meeverhuisd zonder merkbare invloed op de datacommunicatie die erover plaatsvindt. Het aan- en afmelden van nieuwe terminals moet correct worden afgehandeld, evenals de onvermijdelijk optredende crashes van delen van het netwerk.

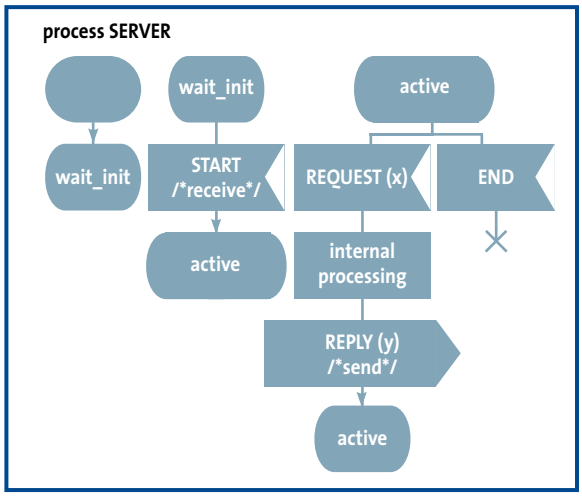
bijvoorbeeld bestaan uit een specifieke input die aan het model wordt aangeboden - de simulatie is geslaagd als het model de verwachte output produceert. Vaak is een te ontwerpen systeem 'reactief', dat wil zeggen dat het een interactie met zijn omgeving onderhoudt, en aldus op voortdurende inputs reageert met voortdurende outputs. Voorbeelden zijn controllers van industriële processen en communicatie-protocollen zoals Mascara.
In dit geval bestaan de scenario's die worden gesimuleerd uit reeksen van stimuli, waarop de verwachte reeks van responsen moet worden vertoond. Niet alleen de waarde of hoedanigheid van dergelijke responsen zijn van belang (bijvoorbeeld een mobiele terminal die buiten het bereik van zijn *access point* raakt moet een nieuw access point kunnen vinden dat alle ATM-kanalen kan overnemen), ook de tijd waarbinnen deze moeten optreden en de relatie met andere reacties maken deel uit van de correctheids-eisen (bijvoorbeeld een dergelijke *handover* moet binnen 100 microseconden worden voltooid).
Voor het opstellen van dergelijke modellen worden zowel 'gewone' programmeertalen (bijvoorbeeld C) gebruikt als specifiekere notaties, die vaak afhangen van het toepassingsgebied. Zo wordt in de telecommunicatie-wereld SDL (system description language) veelvuldig gebruikt. Ook voor het weergeven van scenario's worden formalismen gebruikt, die als voordeel hebben dat ze een eenduidiger betekenis hebben dan bijvoorbeeld natuurlijke taal aangevuld met plaatjes (zie kader SDL en MSC).

Tools

Er is een brede markt voor softwarepakketten die (al dan niet grafische) editors voor zulke notaties integreren met de mogelijkheid tot het uitvoeren van verschillende soorten simulaties, zoals random en geleid. Bij deze laatste vorm heeft de gebruiker de mogelijkheid het verloop van de simulatie te sturen, om zo bijvoorbeeld het systeem in een situatie te manoeuvreren waar fouten worden verwacht.
De mogelijkheden van dergelijke software-gereedschappen zijn gedurende ›

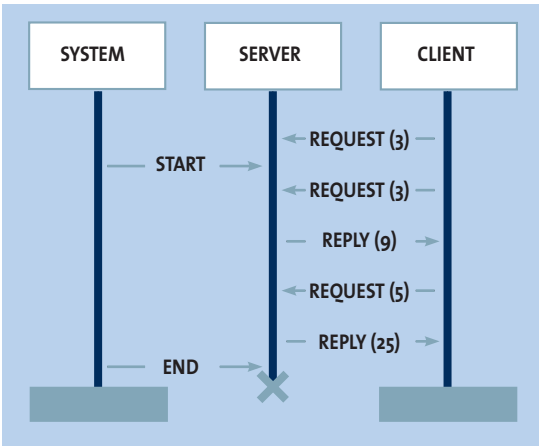
SDL en MSC

SDL (specification and description language) is een taal die specifiek ontwikkeld is voor de beschrijving van telecommunicatie-systemen. Een SDL-systeem bestaat uit afzonderlijke processen die met elkaar communiceren door het sturen van signalen over kanalen. Een proces reageert op de ontvangst van signalen met het doen van interne bewerkingen en daarna verstuuren van nieuwe signalen. Door de grafische notatie kunnen zulke processen inzichtelijk worden weergegeven. Een voorbeeld is te zien in figuur 2.
In MSC (message sequence chart) kunnen specifieke scenario's die in een systeem optreden worden weergegeven. Per proces is er een verticale tijdsas. Signalen corresponderen met pijlen tussen processen (figuur 3). Beide formalismen zijn gestandaardiseerd door de International Telecommunication Union (ITU) en worden ondersteund door diverse software-tools.



2. Een SDL-proces. SDL is een taal die specifiek ontwikkeld is voor de beschrijving van telecommunicatie-systemen. Door de grafische notatie kunnen processen inzichtelijk worden weergegeven.

3. Een *message sequence chart*. In een MSC kunnen specifieke scenario's die in een systeem optreden worden weergegeven.



het laatste decennium sterk toegenomen. Deels is dit natuurlijk te danken aan de explosieve groei in rekenkracht van de computers waar deze tools op draaien. Maar daarnaast zijn er een aantal technieken ontwikkeld die hebben geleid tot een fundamenteel andere benadering van het 'debuggen'. Deze benadering wordt aangeduid met de term model checking.
Net als bij simulatie wordt ook bij model checking een model van het systeem gebruikt om verschillende mogelijke 'runs' te inspecteren. Er zijn echter twee belangrijke verschillen. Ten eerste wordt in het geval van model checking op een specifiek correctheids-criterium getest. Voorbeelden van dergelijke criteria zijn de afwezigheid van *deadlocks* (situaties waarin het systeem vastloopt doordat bijvoorbeeld twee processen op elkaar gaan zitten wachten) en de hierboven genoemde tijdbegrensde responseigenschap dat een handover binnen 100 microseconden voltooid moet worden.
Zo'n criterium wordt geformaliseerd in een zogenoemde *property language*. Zulke talen kunnen lijken op de querytalen die bekend zijn uit de databasewereld (bijvoorbeeld SQL), maar ook bepaalde grafische notaties als MSC of de timing-diagrammen uit de hardware-gemeenschap kunnen in principe hiervoor geschikt zijn.
Het tweede verschil met traditionele simulatie is dat bij model checking alle mogelijke executie-runs van een systeem worden doorlopen. Dit legt sterke beperkingen op aan de grootte van het systeemmodel. Het is dan de verantwoordelijkheid van de ontwerper bij het modelleren zodanige abstracties aan te brengen dat het geheel van beperkte omvang blijft.
Juist doordat er per check meestal een klein aantal correctheids-criteria wordt nagegaan, blijkt dit in praktijk een werkbare benadering. Bijvoorbeeld, de werking van veel communicatie-protocollen is veelal 'data-independent': het protocol kijkt niet wat de inhoud van de te versturen data is. In zo'n geval kan het aantal verschillende berichten dat wordt rondgestuurd worden beperkt tot één of enkele.
Momenteel is er een groeiend aantal tools beschikbaar dat gebaseerd is op deze nieuwe inzichten. De meeste hiervan worden ontwikkeld aan universiteiten of onderzoekslaboratoria van grote bedrijven. Een voordeel hiervan is dat de software meestal vrij beschikbaar is - in een aantal gevallen kan zelfs de broncode worden gedownload, wat de snelle ontwikkeling van dergelijke pakketten ten goede komt.
Een nadeel, althans als het aankomt op toepassing van deze gereedschappen, is dat er vaak sprake is van concentratie op één of enkele aspecten. Elk tool heeft een aantal sterke punten die meestal het bestaansrecht ervan vormen. De integratie van dergelijke aspecten, alsmede het gebruikersgemak (bijvoorbeeld in de vorm van grafische gebruikers interfaces), is lange tijd van ondergeschikt belang geweest. Ook heeft deze verscheidenheid geleid tot een wildgroei op het gebied van modelleringstalen.

Vires

Het Vires-project heeft als doelstelling deze situatie te verbeteren. Enerzijds gebeurt dit door een modelleringstaal zoals SDL te koppelen aan een aantal verschillende model checkers. Centraal hierin staat de notatie IF (intermediate format) die in het project ontwikkeld is als tussenstap in de vertaling van talen als SDL en VHDL naar de specifieke formaten die de verschillende *model checkers* vereisen. Het gebruik van zo'n tussenformaat is een bekende oplossing om het aantal benodigde vertalers te beperken: als zich een nieuwe kandidaat-modelleringstaal aandient die aan de model checkers moet worden gekoppeld, hoeft er slechts één vertaler geschreven te worden, namelijk naar het tussenformaat, in plaats van voor elke tool een aparte vertaler. Anderzijds worden in Vires de verschillende model checkers uitgebreid met nieuwe opties. Deels zijn dit technieken die al worden toegepast in andere checkers en nu, met de nodige aanpassingen en verbe-

teringen, worden geïmplementeerd in de Vires-tools. Deels gaat het om nieuwe technieken die gebaseerd zijn op recent wetenschappelijk onderzoek.
De toetssteen voor deze inspanningen wordt gevormd door Mascara. Hierin zijn een aantal aspecten te vinden die kenmerkend zijn voor embedded systemen en communicatie-protocollen in het bijzonder: realtime gedrag, een sterke afhankelijkheid van stimuli door de omgeving, gelijktijdigheid in de vorm van onafhankelijk lopende, communicerende processen, onderdelen met sterke regelmaat, zoals de aanwezigheid van meerdere (bijna-)identieke processen, alsook grote variaties in datastructuren. Daarnaast is Mascara ook qua omvang representatief voor industriële systemen.
Gedurende de eerste helft van het Vires-project is veel tijd besteed aan het ontwerpen van een Mascara-model in de beschrijvingstaal SDL. In feite betrof dit een reconstructie van (een abstractere versie van) de SDL-specificatie die in het zuster-project Wand is gemaakt. Dit is gebeurd in samenwerking met Wand-partner IntraCom, een Grieks telecombedrijf, om zodoende de getrouwheid van het model te waarborgen. Inmiddels zijn substantiële delen uit dit Mascara-model onderworpen aan de *Vires model checking*-gereedschappen. Op deze wijze zijn diverse bugs aan het licht gekomen. Momenteel concentreert het project zich op het uitbreiden van deze verificatie-experimenten; de resultaten kunnen worden gevolgd op de Vires website (zie hieronder).
Behalve deze resultaten en de diverse uitbreidingen van en koppelingen tussen model checkers, heeft het project een belangrijke spin-off. Zo wordt er in de onderzoeksgroep Informatie en Communicatie Systemen van de faculteit Elektrotechniek van de Universiteit Eindhoven gewerkt aan modellen van Mascara in andere beschrijvingstalen dan SDL. Daarnaast wordt er tegen het einde van Vires een speciale workshop georganiseerd om de resultaten onder aandacht van de industrie te brengen. ◻

Vires

Vires (Verifying industrial reactive systems) is een project dat wordt gefinancierd uit het Europese Esprit-programma. Er wordt samengewerkt door de universiteiten van Kiel (Duitsland), Luik (België) en Eindhoven alsmede het instituut Verimag in Grenoble (Frankrijk). De case study, in de vorm van de specificaties van het Mascara-protocol, is aangeleverd door Lucent Nieuwegein en verder ontwikkeld in samenwerking met IntraCom in Athene. Binnen de Technische Universiteit Eindhoven zijn onderzoekers van zowel de groep Formele Methoden van Informatica als de groep Informatie en Communicatie Systemen van Elektrotechniek werkzaam in het project. De problematiek rond het ontwerp van embedded systemen vraagt om een dergelijke multidisciplinaire aanpak.

Meer informatie op:
radon.ics.ele.tue.nl/~vires/
www.tik.ee.ethz.ch/~wand/