

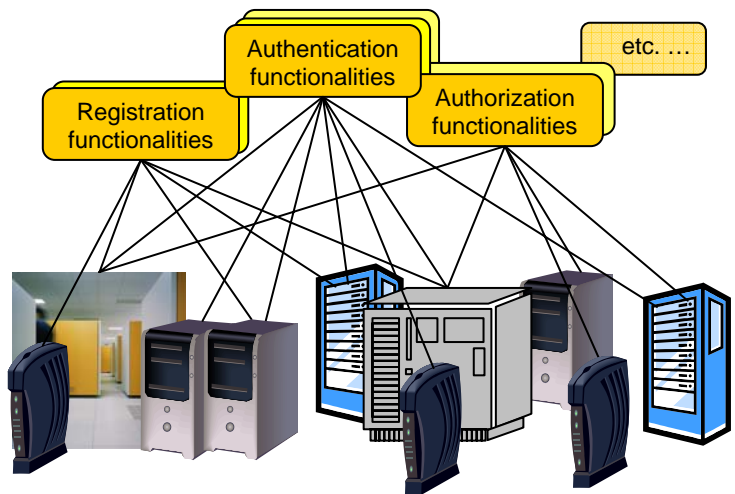
PNP2008 Security Architectuur

voor het leveren van dynamisch geconfigureerde netwerken en diensten

Rieks Joosten
TNO Informatie- en Communicatietechnologie

De markt voor genetwerkte consumentenelektronica en diensten kan alleen groeien als de beloftes van 'eenvoud', 'zelfconfiguratie', 'overal-en-altijd bereikbaar' worden ingelost door de leveranciers van die netwerken, diensten en elektronica. We voorzien marktkansen voor een partij die de bijbehorende vraagstukken van proces- en systeemintegriteit, de integratie ervan en configuratie (personalisatie) oplost. Bedrijven kunnen deze rol nu nog niet aan omdat ze ofwel te klein zijn en de schaalgrootte niet aankunnen, ofwel te groot zijn waardoor de benodigde interne afstemming tussen bedrijfsonderdelen niet rond komt. In het Freeband project PNP2008 experimenteren we met hulpmiddelen voor architectuur dat hierin verandering probeert te brengen.

Het PNP2008 project gaat ervan uit dat mensen in de nabije toekomst zullen beschikken over een aantal eigen apparaten die middels een (persoonlijk) netwerk met elkaar verbonden zijn. De verschillen in deze apparaten en hun onderlinge verbondenheid maakt een wereld aan nieuwe diensten mogelijk. Maar sommige apparaten zijn mobiel. Andere niet altijd beschikbaar. Afhankelijk van plaats, tijd en context kan bijvoorbeeld een afdrukfunctie wel of niet beschikbaar zijn. In zo'n situatie wil een gebruiker niet langer zelf diensten samenstellen tot een geheel, zoals dat bijvoorbeeld nu nog met ADSL, telefonie en digitale TV nodig is. De gebruiker wil dat één partij, die we hier Personal Network Provider (PNP) noemen alle dynamiek, configuraties, klachten, verrekeningen enz. regelt. Deze partij zal dan de specificatie, bouw, levering en uitfasering van diensten, netwerken en apparaten voor zover dat kan, moeten coördineren en automatiseren.



Grote bedrijven bestaan veelal uit onderdelen die resultaatverantwoordelijk zijn gemaakt en die hun eigenbelang meestal boven dat van het geheel stellen. Omdat het voor hen meestal goedkoper is om zelf een systeem te bouwen voor een gegeven functionaliteit, dan om na te gaan of een systeem van een of ander collega-bedrijfsonderdeel kan worden hergebruikt, neemt het aantal systemen steeds maar toe en komt het in grote bedrijven voor dat tientallen systemen gelijksoortige functionaliteit leveren (maar net niet helemaal hetzelfde), en gelijksoortige data opslaan (maar ook weer net niet gelijk). Dit zou je ook verwachten bij een conglomeraat van kleine bedrijven.

Als oplossing hiervoor wordt wel gedacht aan Service Oriented Architectures (SOA's) of Shared Service Centra (SSC's). We zien die nog (te) weinig, onder meer omdat bedrijven het moeilijk vinden de functionaliteiten van hun SOA's of SSC's te uniformeren. Een belangrijke oorzaak hiervoor is dat zo'n uniforme functionaliteit geaccordeerd moet zijn door allerlei partijen, bijvoorbeeld applicatiebeheerders, procesontwerpers en architecten, maar dat elk van deze partijen eigen ideeën heeft over wat bepaalde functionaliteit hoort te doen. Autorisatiefunctie bijvoorbeeld is voor de één 'het uitdelen van een permissie om iets te mogen doen' en voor een ander 'het controleren van een permissie om iets te mogen doen'. Willen we een eenduidige, uniforme functionaliteit vastleggen, dan is het noodzakelijk noties als 'klant', 'dienst', 'netwerk' enz. ook eenduidig en uniform vast te leggen.

In het Freeband project PNP2008 maken we voor de PNP architecturen met daarin drie lagen (zie onderstaande figuur). In de conceptuele laag leggen we in gezamenlijk overleg de te gebruiken terminologie eenduidig vast (zie kader).

Nadat de betrokkenen zich hieraan verbonden hebben wordt een functionele architectuur gemaakt. Deze bestaat ondermeer uit een verzameling functies en het bijbehorende datamodel. Echter, omdat de keuze van de functies precies komt – ze moeten namelijk bruikbaar zijn in een veelheid aan use-cases – is deze klus lastiger dan op het eerste gezicht lijkt. Het systematisch uitwerken en documenteren van deze use-cases, daarbij gebruikmakend van de vastgelegde terminologie, geeft sturing als zulke functionele keuzes moeten worden gemaakt.

Als de functionele view eenmaal is vastgesteld, is het realiseren van één of meer instanties van deze functionaliteit – de realisatie of systeem view – niet zo spannend meer.

Een PNP architectuur moet niet alleen worden gemaakt, maar ook onderhouden en beheerd om efficiënt te kunnen reageren op veranderingen in de markt, in de organisatie, enzovoorts. Het is dan belangrijk dat nieuwe situaties kunnen worden vergeleken met bestaande views, en dat aanpassingen relatief gemakkelijk kunnen worden gemaakt met minimale impact. In de beschreven drie-lagen benadering kunnen veranderingen in de realisatie laag eenvoudig worden gerealiseerd zolang de functionaliteit maar onveranderd blijft. Als die echter toch verandert, heeft dit alleen impact op het handje vol systemen dat die functionaliteit implementeert en als de nieuwe functionaliteit niet met de oude compatibel is, ook op de applicaties die de oude functionaliteit gebruiken. Het is de bedoeling dat de conceptuele view nog minder frequent verandert. Immers, als die verandert betekent dit dat het bedrijf op een andere manier naar de wereld wenst te kijken. Dat kan gebeuren maar moet niet vaak voorkomen mede gezien de grote impact die dit heeft. Echter, met voldoende gereedschap kan de architect de scope van de impact precies bepalen en daarmee de impact beperkt houden.

De effecten die we van deze aanpak verwachten zijn noodzakelijk voor PNP's in de markt, en zijn herleidbaar op de complexiteitsreductie die vooral het maken van een goede conceptuele view met zich meebrengt. Met behulp van een tool kan een architect of ontwerper controleren of het ontwerp consistent is met de conceptuele view. Een uitbreiding op die tool kan (real-time) de operationele systemen toetsen, en dit gaat leiden tot de forse toename in integriteit van ontwerp en realisaties, welke de PNP nodig heeft om enerzijds flexibel en anderzijds op robuuste wijze diensten op de persoonlijke netwerken van de toekomst te kunnen leveren.

Rieks Joosten (rieks.joosten@tno.nl) houdt zich als onderzoeker/consultant bij TNO Informatie- en Communicatietechnologie bezig met informatie-beveiliging en architectuur. In het bijzonder gaat zijn aandacht uit naar integriteit van architectuur, processen, systemen en het ontwerpen daarvan. Eerder publiceerde hij daarover onder meer in Telecommagazine en Computable.

De Conceptuele View van een architectuur bestaat uit concept(namen), relaties daartussen en regels. Deze worden uitgedrukt zowel in natuurlijke taal als in relationele algebra (of predikatenlogica). Hiermee wordt de semantiek van de concepten eenduidig begrensd, en wordt het mogelijk consistentie te toetsen.

Neem bijvoorbeeld de regels: 'Elk product wordt pas geleverd als een order voor dat product is geaccepteerd' en 'Als een product is geleverd, dan is er een rekening gestuurd naar de klant die de order heeft geplaatst en op die rekening is het product gefactureerd'. Uit deze regels volgt dat een product iets is dat je kan leveren, je kunt er orders voor plaatsen en je kunt het factureren. Onder deze regels is een dienst dus een product, omdat je elke dienst kunt leveren, ordenen en factureren. Alles wat je niet kunt leveren kan geen product zijn. Alles wat je niet kunt factureren of ordenen kan geen product zijn.

Op deze manier wordt elk concept vastgelegd, niet door het te definiëren maar door criteria vast te leggen waaraan iets moet voldoen wil het gezien kunnen worden als voorbeeld van zo'n concept.

