

Comunicacions Grupals en el Programari Lliure

Andreu Moreno¹ i Joan Valduvico^{2,3}

¹ Escola Universitària Salesiana de Sarrià (EUSS). St. Joan Bosco, 74. 08017 Barcelona. <http://www.euss.es>

² LAIGU,SCCL. Barcelona <http://www.laigu.net>

³ Cirsa Interactive Corp. Terrassa.

Correu: amoreno@euss.es, joan@laigu.net

Resum

Les Comunicacions Grupals són un dels sistemes bàsics per implementar sistemes distribuïts que requereixin certes garanties en el subministrament de la informació i un control instantani de les màquines disponibles. El projecte dFSM presentat anteriorment, ha aportat una plataforma que permet dissenyar un sistema com un conjunt d'autòmats finits que s'executen concurrentment en una mateixa màquina. Ara se'n presenta una ampliació en el suport de sistemes basats en autòmats que s'executen en un sistema distribuït. S'ha realitzat una recerca i selecció de sistemes de comunicacions grupals dins de l'àmbit del programari lliure, el resultat del qual has estat Spread i Ensemble. Finalment només el primer ha complert tots els requeriments per afrontar el suport de sistemes distribuïts per la dFSM. Aquesta comunicació presenta precisament l'adaptació de la dFSM per suportar Spread. Tot el codi desenvolupat és accessible des del portal SourceForge (<http://dfsm.sourceforge.net>).

Introducció

El projecte dFSM [1] neix per oferir una plataforma de disseny i execució d'aplicacions encastades, que en la majoria des casos estan basades en màquines d'estats finites. La idea és oferir al programador un sistema que permeti de forma molt simple codificar una aplicació com un conjunt d'autòmats que s'executen de forma concurrent en una mateixa màquina. Adicionalment el sistema dFSM adopta com a mecanisme bàsic de comunicacions un sistema d'events, que es subscriuen prèviament els autòmats i en ser rebuts causen l'evolució del seu estat. Aquest projecte continua el treball en el sistemes que poden implementar-se com autòmats distribuïts fent possible que els diferents autòmats puguin ara estar repartits per les diferents màquines d'un sistema distribuït. Per tal de disposar d'aquesta funcionalitat s'ha optat per basar l'adaptació en un sistema de Comunicacions Grupals[2]. Tot aquest treball s'emmarca en un gran projecte en el que es pretén el desenvolupament d'una plataforma per desenvolupar sistemes tolerants a fallades basats en autòmats distribuïts inspirada en el treball de Schneider[3].

En aquesta comunicació es presenta en primer lloc una introducció als sistemes de Comunicacions Grupals, a continuació es realitza una recerca del sistemes accessible actualment i se'n fa un anàlisi des de l'òptica del programari lliure. Finalment es presenta l'adaptació que s'ha realitzada de dFSM sobre el sistema Spread [4] i les conclusions.

Comunicacions Grupals

Els sistemes distribuïts constitueixen una de les principals opcions per assolir tolerància a fallades amb la introducció de redundància. La programació d'aquests sistemes no és simple, especialment si es té present que cal garantir consistència en la informació replicada front les fallades del sistema.

Les comunicacions grupals [2] s'han consolidat com una de les opcions per assolir tolerància a fallades gràcies als serveis d'enviament dels missatges a grups de processos amb certes garanties. El processos del sistema s'organitzen en grups que es comuniquen mitjançant l'enviament de missatges a un grup. Internament el missatge es fa arribar a tots els integrants (anomenat multienviament) amb garanties d'entrega i d'ordenament. En tot moment es manté la llista dels integrants del grup que són accessibles i s'informa quan hi ha incorporacions o baixes a tots els membres actuals del grup (gestió de membres), és el que s'anomena vista del grup.

El multienviament dels missatges es pot realitzar amb diferents modes pel que fa a l'ordre i la garantia en el subministrament als processos destinataris. Es distingeix entre rebre un missatge (rebut pel sistema de

comunicacions grupals) i subministrar un missatge al procés destinatari. Així el disposa principalment dels següents modes de multienviament:

- Mode causal. Es garanteix que l'ordre dels missatges preserva la relació *ocorregut abans* (*abans* de Lamport [5]). Significa que missatges generats pel mateix procés sempre són subministrats en el mateix ordre en que foren enviats a tots els membres del grup, i que una resposta a un missatge mai es subministra a un tercer abans que el missatge que l'ha causat. En conseqüència els missatges que es generen en processos diferents sense cap tipus de relació causal, podran arribar als diferents membres del grup en ordre diferents. El sistema de comunicacions grupals s'encarrega d'encuar el missatges un cop rebuts i només subministra-los quan els criteris del mode ho permetin.
- Mode ordre total. En aquest cas es tracta de garantir l'ordre total dels missatges, és a dir, completar el model causal amb el cas de missatges generats en processos diferents sense cap relació causal. Evidentment que aquesta garantia és a costa d'una penalització en el temps de subministrament.
- Mode segur. Aquest mode garanteix que no es subministra el missatge als membres del grup fins que es tingui constància que ha estat rebut per tots els membres. Un cop els missatges són rebuts pels membres, s'envia una missatge de reconeixement indicant que s'ha rebut i quan tots els membres del grup són conscients que tots l'han rebut, llavors es subministra als processos destinataris. Pel que fa a l'ordre de subministrament, es tenen les mateixes garanties que en Mode ordre total.

Analitzant el comportament dels diversos modes amb més profunditat sorgeix la pregunta de què passa si un dels integrants ha patit algun tipus d'anomalia i no rep el missatge o no pot enviar el missatge de retorn (en el cas del mode segur), llavors es quan entra en acció el servei de gestió de membres, el qual informa als membres restants del grup la nova composició (envia la nova vista del grup) i tot seguit es subministra el missatge pendent si el mode escollit ho ha evitat fins ara. Així les aplicacions acaben rebent el missatge i a més tenen constància quins processos l'han rebut.

Els sistemes de comunicacions grupals acostumen a oferir a les aplicacions un entorn anomenat Sincronia Virtual [6], el qual fou introduït en primer lloc dins del projecte ISIS [7]. Pretén definir la relació entre els missatges ordinaris i els missatges de control que informen de canvis en la gestió de membres en el grup (vistes). En essència la Sincronia Virtual estableix la garantia de que el canvis en la gestió de membres són observats en el mateix ordre per tots els processos que resten connectats. Això suposa que els canvis en la gestió de membres estan totalment ordenats respecte els missatges ordinaris, independentment del mode que estiguin emprant. Per tant dos processos que segueixin operant correctament observen el mateixos canvis consecutius en el membres accessibles i reben el mateix conjunt de missatges ordinaris entre els canvis.

El sistema ISIS[7] únicament contemplava un model de malfuncionament en el que un procés si experimenta una errada es para automàticament. En el cas de considerar un model més ampli, en el que un procés pugui experimentar alguna anomalia però que continuï operant de forma separa (degut per exemple a un problema de comunicacions), apareix el que s'anomena partició, llavors l'entorn s'amplia amb el concepte de Sincronia Virtual Estesa [8], que afegeix que en cada una de les particions que es puguin formar es mantindrà la Sincronia Virtual de forma aïllada.

En un entorn el que es contempen les particions es defineix el concepte de Partició Primària com la formada per la majoria dels processos. Així es dona un suport bàsic al cas en que les particions es puguin tornar a fusionar quan l'anomalia s'ha solucionat (per exemple restabliment de les comunicacions) informant de la nova composició del grup amb l'enviament d'una nova vista. Això permet que la Partició Primària pugui sincronitzar els processos de les particions fusionats mitjançant un procés de tornar a transferir els missatges perduts i així es restableixi l'estat.

Comunicacions Grupals en el Programari Lliure

Les comunicacions grupals han estat un camp de recerca força prolífic en els darrers anys, amb un moment de destacat esplendor a finals del noranta. Des de la publicació del sistema ISIS[7], considerat el primer, han aparegut en escena una desena de sistemes provinents de diversos grups de recerca d'arreu. En la majoria dels sistemes únicament es pot accedir a les publicacions presentades, però el codi no pot aconseguir-se. Excepcionalment això es possible en alguns casos amb la signatura d'un document on un s'ha de comprometre a utilitzar-lo únicament amb finalitats de recerca i mai amb fins comercials. Afortunadament hi ha dos sistemes que s'han emmarcat dins del món del programari lliure: Ensemble [9] i Spread [4]. Tots dos generats en consolidats grups de recerca amb una llarga tradició en el camp de les comunicacions grupals.

Una recerca dels més destacats sistemes de comunicacions grupals evoca la següent llista [10]: Consul, Ensemble, Horus, Isis, Newtop, Phoenix, Relacs, Spread, Timewheel, Totem, Transis i xAMp. Els autors d'aquest treball només han aconseguit els fonts de Ensemble, Horus, Spread i Transis. A continuació es presentarà cada un dels casos amb una breu descripció de l'entorn en que foren gestats.

Transis

Transis és un sistema de comunicacions grupals que ha estat desenvolupat pel *High Availability Laboratory* del *Computer Science Department* de la *Hebrew University* de Jerusalem, sota el guiatge de Danny Dolev i Dahlia Malkhi. Aquest grup de recerca ha ocupat un dels llocs destacats en la recerca en aquest camp.

Transis [11] constitueix una capa de comunicacions multienviament que facilita el desenvolupament d'aplicacions distribuïdes tolerants a fallades. Disposa de serveis de comunicacions fiables per aplicacions que requereixen alta disponibilitat i un conjunt de protocols que assoleixen altes prestacions amb el maquinari existent gràcies a la eficiència del protocol Trans [12]. També permet gestionar la consistència en casos de particionament i en la fusió de grups.

Per tal de tenir accés al codi font de transis, es necessari signar una llicència on l'usuari es compromet a utilitzar Transis únicament per a recerca i mai amb finalitats comercials. Aquesta condició i el fet que disposem d'alternatives ha motivat que aquest sistema no es considerés una de les opcions per donar servei de comunicacions grupals al sistema dFSM.

Horus i Ensemble

Horus i Ensemble [9] són dos sistemes de comunicacions grupals desenvolupats al *Department of Computer Science* de la *Cornell University*, sota la direcció de Ken Birman. Aquest grup de recerca s'ha de considerar el bressol de les comunicacions grupals. Ells van obrir camí amb el primer sistema, el ISIS, el qual de seguida es va consolidar com sistema utilitzat per implementar sistemes distribuïts per moltes companyies a finals dels anys vuitanta. A més amb aquest projecte es van establir les bases teòriques sobre les comunicacions grupals, especialment tot el referent a la sincronia virtual [6].

Amb el temps es van veure abocats a fer un replantejament més modular, degut a que tenien molts problemes per mantenir la qualitat del codi en les successives ampliacions. Aquí és gesta en la primera meitat dels anys noranta el Horus, sistema que es caracteritza per estar compost per una arquitectura basada en una pila de capes pràcticament independents, cada una encarregada de oferir una determinada característica al sistema, i fent possible que configurar el sistema sigui simplement escollir un conjunt de capes.

Els requeriments cada cop més exigents al sistema Horus van fer que tot i la millora amb el pas d'un sistema monolític a un de modular, el sistema cada cop era més complex de mantenir, especialment motivat per limitacions del llenguatge de programació utilitzat, el C. Tot plegat va fer que l'any 1996 s'iniciés el desenvolupament del sistema Ensemble, el qual n'hereta l'arquitectura modular però està implementat amb un llenguatge d'alt nivell, la variant O'Caml del llenguatge ML [13]. Aquest canvi els va permetre un cert grau d'automatització del procés de codificació assolint amb menys temps el mateix nivell de qualitat [9].

Es pot disposar del codi tant del projecte Horus com Ensemble. En el cas de Horus, cal signar una declaració d'ús únicament per finalitats de recerca. En canvi Ensemble està publicat sota una llicència lliure tipus BSD. El projecte Horus no s'està mantenint des del l'any 96 i per tant el seu ús no es recomana, en canvi Ensemble està mantenint actualment per part de Ohad Rodeh de IBM i disposa d'una certa comunitat d'usuaris establerta en una llista de distribució.

Spread

Spread [4] és el sistema de comunicacions grupals desenvolupat al *Department of Computer Science* de la *Johns Hopkins University*, sota la tutela de Yaig Amir. Es dona la casuística de que el Dr. Amir va realitzar la tesi doctoral a la *Hebrew University* de Jerusalem col·laborant estretament en el desenvolupament del sistema Transis[11] i va realitzar una estada al *Department Electrical and Computer Engineering* de la *University of California* on es desenvolupava el sistema Totem [14] amb una contribució important en els protocols d'ordenació de missatges[15]. Per tant poder afirmar que en Dr. Amir és un dels científics més autoritzats en la matèria.

El sistema Spread és molt similar al sistema Transis, però amb l'avantatge de que està editat amb una llicència tipus BSD i que disposa actualment d'una comunitat d'usuaris i desenvolupadors molt activa (molt més que la d'Ensemble)

Adaptació de la dFSM per un sistema distribuït

De l'anàlisi realitzat en el apartat anterior se'n desprèn que en un marc de programari lliure, les úniques opcions són Ensemble i Spread. A més respon afortunadament als casos en que hi ha establert una comunitat que manté el programari. Partint d'aquesta base, en primera instància el projecte dFSM va intentar adaptar-se tant per Ensemble com Spread.

El projecte dFSM[1] permet desenvolupar una aplicació com un conjunt d'autòmats que s'executen concurrentment en una mateixa màquina amb un mecanisme de comunicacions basant en l'enviament d'events, la recepció dels quals provoca el canvi d'estat en els autòmats. Ara es tracta d'anar un pas més enllà i fer possible que el conjunt d'autòmats puguin estar repartits en les diferents màquines d'un sistema distribuït. Perquè això sigui possible cal disposar d'un sistema de comunicacions distribuïdes. Aquí és on entra en escena les comunicacions grupals. Les requeriments necessaris per fer-ho possible són:

1. Els sistemes de comunicacions grupals estan basats en l'abstracció de grup, en el sentit que el mecanisme d'enviament va adreçat a enviar un missatge a tots els integrants d'un grup. En el cas del sistema dFSM, els destinataris d'un event són tots els autòmats que tinguin una transició entre estats que en depenguin. Es tracta doncs de definir un grup per cada un dels events que s'utilitzen.
2. Les garanties amb que es realitza el subministrament del missatge són d'especial importància. No és permisible que dos missatges es subministrin en ordre diferents en autòmats diferents. Aquesta situació comportaria inconsistències en l'estat dels autòmats i afecta en general el determinisme del comportament. Aquest problema és més evident en el cas extrem d'utilitzar el sistema per replicar els autòmats en un sistema distribuït, és evident que alteracions en l'ordre de subministrament comportaria que l'autòmat replicat tingués estats diferents en funció de la màquina.

El principi de disposar de suport pels sistemes Spread i Ensemble, perseguia que la doble adaptació aportaria un coneixement molt més sòlid sobre les comunicacions grupals. A l'hora de concretar s'ha arribat a la conclusió que de moment es desestimava l'opció de Ensemble. La raó la trobem en el fet que aquest sistema obliga a que per tal que un procés pugui enviar un missatge a un grup, l'emissor ha d'estar registrat i per tant també rep aquest missatge. Actualment en la dFSM únicament es registren els receptors de missatges, pel que fa a l'enviament, no tenia sentit. Evitar modificar aquesta interfície suposava que el primer cop que un autòmat envia un missatge havia de registra-lo i a més llavors cal esperar rebre un missatge que confirma que ja es forma part del nou grup per procedir a enviar. Aquesta limitació suposa una penalització important en el primer cop que s'envia un event. En el cas de Spread (sense aquesta limitació), l'adaptació s'ha completat amb èxit i les diferents proves experimentals han permès demostrar la viabilitat d'aquesta opció per fet la dFSM un sistema distribuït.

Conclusions

En aquest treball s'ha presentat una visió general del sistema de comunicacions grupals com un dels mecanismes disponibles per implementar un sistema distribuït. Ademés s'ha presentat un petit anàlisi des del punt de vista del programari lliure, el qual arriba a la conclusió les opcions més sòlides són Ensemble i Spread. El fil conductor ha estat l'ampliació de les prestacions del sistema dFSM per tal que els diferents autòmats que constitueixen una aplicació puguin estar distribuïts en els diferents nodes d'un sistema distribuït. Finalment problemes concrets en la implementació han fet finalment que només l'opció de dFSM fos viable. Actualment s'ha codificat i verificat aquesta opció i forma part del codi de la dFSM (<http://dfsm.sourceforge.net>).

Aquest projecte s'emmarca en un gran projecte en el que es pretén el desenvolupament d'una plataforma per desenvolupar sistemes tolerants a fallades basats en autòmats distribuïts. El desenvolupament del sistema dFSM i la posterior ampliació a sistemes distribuïts han estat les primeres passes per disposar d'un sistema amb tolerància a fallades sota certes models de fallada del sistema.

Agraïments

El sistema dFSM utilitza programari desenvolupat per Spread Concepts LLC per ser utilitzat en l'eina Spread. Per més informació sobre Spread veure <http://www.spread.org>.

Referències

- [1] A. Moreno and J. Valduvico. *dFSM: Finite State Machines for Embedded Systems*. V Real Time Linux Workshop. p. 239-245. Valencia, 2003.
- [2] D. Powell. *Group Communication*. Communications of the ACM. 39(4). p. 50-53. 1996.
- [3] F. B. Schneider. *Implementing Fault-Tolerant Services Using the State Machine Approach: A Tutorial*. ACM Computing Surveys. 22(4). p.299-319. 1990.
- [4] Y. Amir and J. Stanton. *The Spread Wide Area Group Communication System*. TR CNDS-98-4, The Center for Networking and Distributed Systems, The John Hopkins University. 1998
- [5] L. Lamport. *Time, Cloks and the Ordering of Events in Distributed Systems*. Communications of the ACM. 21(7). p. 558-565. 1978.
- [6] K. Birman and T. Joseph. *Exproiting virtual synchrony in distributed systems*. 11th ACM SIGOPS Symposium on Operating Systems Principles (SOSP). p. 123-138. 1987
- [7] K. Birman and R. van Renesse. *Reliable Distributed Computing with the Isis Toolkit*. IEEE Computer Society Press. 1994
- [8] L.E. Moser, Y. Amir, P.M. Melliar-Smith, D.A. Agarwal. *Extended Virutal Synchrony*. 14th International Conference on Distributed Systems (ICDCS). p. 56-64. 1994.
- [9] K. Birman, B. Constable, M.Hayden, J. Hickey, C. Kreitz, R. van Renesse, O. Rodeh and W. Vogels. *The Horus and Ensemble Projects*. DARPA Information Survivability Conference & Exposition (DISCEX '00), p. 25-27. 2000.
- [10] Gregory V. Chockler. *Group Communication Specifications: A Comprehensive Study*. ACM Computer Surveys. 33(4). p. 1-43. 2001.
- [11] Y. Amir, D. Dolev, S. Kramer and D. Malki. *Transis: A communications subsistem for high availability*. 22nd IEEE Fault-Tolerant Computing Symposium (FTCS). 1992
- [12] P.M. Melliar-Smith and L.E. Moser. *Trans: A reliable broadcast protocol*. IEE Trans. Commun. 140 (6) , p. 481-493. 1993.
- [13] X. Leroy. *The Objective Caml System Release 1.05*. INRIA. France, 1997.
- [14] L.E. Moser, P.M. Melliar-Smith, D.A. Agarwal, R.K. Budhia and C.A. Lingley-Papadopoulos. *Torem: a Fault-Tolerant Multicast Group Communication System*. Communications of the ACM. 39(4). p. 54-63. 1996.
- [15] Y. Amit, L.E. Moser, P.M. Melliar-Smith, D.A. Agarwal and P. Ciarfella. *The Totem Single-Ring Ordering and Membership Protocol*. ACM Transactions on Computer Systems. ??????