Gestión de cambios de configuraciones de TI con Coloured PetriNets

Sergio Machuca, Gabriela A. Sasco

Telemática, Eduardo Acevedo 1622
11200 Montevideo, Uruguay
{smachuca, gsasco}@telematica.com.uy
http://www.telematica.com.uy

Abstract. En los últimos años las organizaciones han aumentado su dependencia de las tecnologías de la información (TI). Las infraestructuras utilizadas incluyen gran número de componentes (aplicaciones, servidores, bases de datos, componentes de comunicaciones, UPS, etc.), requiriendo una gestión cuidadosa.

COBIT e ITIL plantean la necesidad de mitigar los riesgos de esa dependencia. Coinciden que la gestión de TI debe centrarse en los servicios y que es necesaria una base de datos de configuraciones (CMDB) para conocer los componentes de infraestructura; sus dependencias; usuarios; conexiones eléctricas y de red, etc. Esta CMDB permite optimizar la gestión de cambios e incidentes, y apoyar la gestión de disponibilidad, capacidad, continuidad y acuerdos de niveles de servicio.

Mostraremos como modelar los componentes de la infraestructura de TI utilizando PetriNets. Se analiza como representar diversas configuraciones y se utilizan Coloured PetriNets para algunas configuraciones específicas. También mostramos como realizar análisis de impacto de los servicios.

Keywords: ITIL, CIBIT, Coloured Petri Nets, Gestión Configuración, Gestión Cambios.

1 Introducción

En los últimos años, la utilización de las tecnologías de la información (TI) en las organizaciones se ha incrementado significativamente llegando a requerir infraestructuras muy complejas. Las necesidades de negocio requieren nuevas tecnologías y aumentan la cantidad de aplicaciones, servidores, sistemas operativos, bases de datos, dispositivos de comunicaciones, generadores de energía, etc.

Esta mayor dependencia de la tecnología aumenta los requerimientos de disponibilidad y niveles de servicio que ocasionan que cualquier falla o baja de un servicio, si no es conocida de antemano o es muy prolongada, se convierta es una

situación catastrófica para la organización. Para mitigar estos riesgos, se deben adoptar las medidas necesarias para asegurar que los tiempos de indisponibilidad causados por fallas y los ocasionados por cambios de configuraciones en la infraestructura de TI sean minimizados.

COBIT [1] e ITIL [2] introducen la necesidad de la gestión de los servicios de TI y plantean procesos como Configuration Management, Availability Management, Capacity Management, Service Level Management, IT Service Continuity Management, etc.

Ninguna de ellas especifica como modelar la infraestructura de TI, ni como realizar análisis de impacto. No existe bibliografía que trate este tema específicamente, pero existen diversas referencias sobre gestión de cambios en componentes de software.

En trabajos anteriores [3],[4],[5],[6] vimos la necesidad de contar con una base de datos de configuraciones (CMDB) para conocer todos los componentes de TI; las dependencias entre ellos; usuarios y administradores; conexiones eléctricas y de red, etc. Esta CMDB es el centro de la gestión de los servicios basados en ITIL. Representar la infraestructura de TI utilizando Abstract System Dependence Graph (ASDG) [7] y mostramoscomo gestionar los cambios en los componentes de infraestructura. Con ASDG no pudimos resolver completamente el problema. No pudimos representar las conexiones de red ni modelar, sin adaptaciones, configuraciones complejas como las de alta disponibilidad, clusters, etc. Otro tema no contemplado es el tratamiento del tiempo.

En este trabajo mostramos como utilizar PetriNets [8], para modelar la infraestructura de TI. El artículo se organiza como sigue: primero se presentan los principios básicos de la gestión de la configuración y luego se muestra como modelar la infraestructura utilizando PetriNets. Se analizan diferentes situaciones incluyendo algunas de las configuraciones que no se pudieron representar con ASDG. También se muestra la necesidad de utilizar Coloured PetriNets [16] para representar algunas configuraciones. Se presenta un ejemplo de una infraestructura modelada con PetriNets. Finalmente en las conclusiones comentamos los resultados obtenidos y los trabajos complementarios que hemos iniciado.

2. Gestión de la configuración

La necesidad de una gestión de las configuraciones es realizada en propuestas tales como COBIT, ITIL y Microsoft Operations Framework (MOF) [9], entre otras.

Las metas definidas por COBIT [1] son “dar cuenta de todos los componentes de IT, prevenir alteraciones no autorizadas, verificar la existencia física y proporcionar una base para el sano manejo de los cambios”.

COBIT hace énfasis en la orientación al negocio y ha sido diseñado no solo para ser utilizado por usuarios y auditores, sino que en forma más importante, está diseñado para ser utilizado como una lista de verificación detallada para los propietarios de los procesos de negocio. En el Marco Referencial de COBIT se proporcionan herramientas al propietario de procesos de negocio que facilitan el cumplimiento de sus responsabilidades. El marco referencial comienza con la premisa “Con el fin de proporcionar la información que la empresa necesita para alcanzar sus

objetivos, los recursos de TI deben ser administrados por un conjunto de procesos de TI agrupados en forma natural".

Por otro lado la visión de ITIL [2], [9], [10], [11], [12], [13] es: "La gestión de la configuración es la implementación de una base de datos (Configuration Management Database – CMDB) que contiene detalles de los elementos de la organización que son usados en la provisión de servicios de IT. Esto es más que solo un registro de activos, pues debe contener información de mantenimiento, movimiento y problemas experimentados por los componentes". MOF [14] es una propuesta de Microsoft basada en ITIL que podemos considerar que comparte esta visión.

La gestión de los niveles de servicio es la base del planteo de ITIL. Al analizar sus procesos se observa que la gestión de las configuraciones es la que brinda, a través de su CDMB (Configuration Management Data Base), la información necesaria para la mayoría de los procesos, pues permite conocer la infraestructura propiamente dicha.

A grandes rasgos podemos ver que ambas propuestas, al igual que otras menos difundidas [15], plantean la necesidad de identificar, controlar y mantener todos los componentes de IT. En una gran organización esta estructura llega a tener miles de componentes y se hace imprescindible un manejador de bases de datos.

Entre los componentes se deben incluir:

- Componentes eléctricos (generadores, UPS, cajas de distribución, etc.).
- Dispositivos de comunicaciones (routers, switchs, hubs, firewalls, etc.).
- Hardware (servidores, tarjetas de red, fuentes, periféricos, etc.).
- Software (sistemas operativos, servidores de bases de datos, servidores de aplicaciones, aplicaciones propias, etc.).
- Personas (usuarios, administradores, etc.).
- Relaciones de dependencia (aplicación depende de servidor de aplicación, aplicación depende de repositorio de base de datos, servidor necesita UPS, etc.).
- Conexiones de comunicaciones (hub conectado a switch, servidor conectado a hub, switch conectado a router, etc.).

A pesar de la existencia de una CMDB, el problema de realizar un cambio en la configuración es complejo pues necesitamos realizar una recorrida por todas las dependencias y conexiones para encontrar todos los componentes, usuarios y administradores involucrados.

3. Metodología propuesta

El problema que se desea resolver es conocer el impacto (usuarios afectados, administradores, aplicaciones, servicios, conexiones, etc.), cuando un componente de la infraestructura (red, hardware, software de base, software de aplicación, eléctrico, etc), deje de funcionar ya sea por un cambio programado o por falla del mismo.

Para resolverlo, se necesitan conocer todos los componentes de infraestructura de TI, como están conectados, de que otros componentes dependen, quienes son los usuarios de los servicios, quienes son los administradores de los distintos componentes, etc.

Lo anterior implica representar la interconexión de componentes (Ej. aplicación que corre en determinado servidor de aplicación, aplicación que utiliza determinado manejador de bases de datos, equipo conectado a cual UPS, etc.).

3.1. Especificación del problema

Vamos a representar el problema visualizando la infraestructura de TI como una PetriNet [8], donde los servidores, dispositivos de red, software de base, aplicaciones, dispositivos eléctricos y usuarios, son nodos (Places) y sus relaciones de dependencia, conexiones físicas y eléctricas, son conexiones (transitions) entre ellos.

Analizar el impacto de un cambio o falla de un componente se traduce en la PetriNet a encontrar todos los nodos alcanzados en una ejecución de la red a partir de marcar los nodos origen del cambio o falla.

Veremos como modelar los distintos tipos de configuraciones, incluyendo algunos casos complejos que no pudimos resolver con ASDG.

3.2. PetriNets

Una red es definida formalmente como una tripleta N = (S, T, F) tal que:

- S y T son conjuntos disjuntos (los elementos de S son llamados S-elementos y los de T, T-elementos)
- $F \subseteq (S * T) \cup (T * S)$ es una relación binaria

Una marcación de una PetriNet es una asignación de tokens a los S-elementos y esta define el estado de una red. El número y posición de estos tokens puede cambiar durante la ejecución de la red.

La ejecución es controlada por el número y distribución de los tokens. Se ejecuta disparando transiciones y una transición se dispara removiendo tokens de los S-elementos de entrada y creando nuevos en los de salida, pudiendo ser disparada cuando se encuentra habilitada, es decir cuando en cada S-elemento de entrada hay al menos un token por cada arco de entrada de la transición.

A partir de esto, es posible definir diversos tipos de interpretaciones de las PetriNet, permitiendo con ellos especificar colas y tokens con identidad, de manera de simplificar la cantidad de Places y Transiciones, así como también la posibilidad de especificar tiempos.

Place/Transition nets (P/T nets).

En esta clase de redes de Petri se llaman «Places» a los S-elementos y «Transitions» a los T-elementos. Son las comúnmente llamadas PetriNets y básicamente es a partir de estas que surgen las distintas extensiones.

Coloured PetriNets (CPN) [16]

Son una extensión de las anteriores en que las marcas pueden pertenecer a un tipo determinado (coloreado), lo cual permite diferenciar distintas marcas.

Las principales diferencias con las P/T Nets son las siguientes: los "Places" de una CPN pueden contener marcas de un tipo determinado, los arcos de una CPN poseen expresiones de arco y las transiciones pueden tener una condición de habilitación.

3.3. Solución al problema

Mostraremos ahora como representar con PetriNets, los componentes de la infraestructura, sus conexiones y dependencias.

Para esto, consideremos los componentes como "S-elementos" de nuestra PetriNet y las dependencias como "T-elementos" de la misma.

Por consiguiente, definamos formalmente nuestra PetriNet de la siguiente manera:

$$\mathbf{PN = (S=C \cup RRHH, T, FF)} \quad (1)$$

donde

C son los componentes de la infraestructura

RRHH son recursos humanos de la organización

T y **FF** son las transiciones que representan las relaciones entre los componentes

A continuación se detallarán los mismos.

Sea **C (componentes de la infraestructura) = F ∪ L ∪ E ∪ O**

F es el conjunto de componentes físicos

L es el conjunto de componentes lógicos

E es el conjunto de componentes eléctricos

O es el conjunto de otros componentes

F (componentes físicos) = N ∪ H ∪ OTF

N = HUB ∪ SWT ∪ ROU ∪ FRW

HUB = hubs; SWT = switchs; ROU = routers y FRW = firewalls

H = EQP ∪ P, con P = IMP ∪ SCN ∪ MON

EQP = Hardware (Servidores, PCs); IMP = impresoras; SCN = scanners y MON = monitores

OTF (otros componentes)

L (componentes lógicos) = SO ∪ SRV ∪ SFTB∪ APL ∪ OTL

SO = sistemas operativos; SRV = servicios; SFTB = software de Base; APL = aplicaciones; OTL otros componentes lógicos

E (componentes eléctricos) = UPS ∪GEN ∪ CDD ∪OTE

UPS; GEN = generadores; CDD = cajas de distribución; OTE = otros componentes eléctricos

Los **recursos humanos** se definen de la siguiente manera: **RRHH = ADM ∪ DES ∪ USU**

ADM el conjunto que representa los administradores de recursos

DES el conjunto que representa los desarrolladores

USU el conjunto que representa los usuarios de un servicio

T es el conjunto de transiciones de la Red de Petri, y FF representa las relaciones entre los "S" elementos y los "T" elementos. Ambos nos permiten modelar las relaciones entre componentes.

Para realizar un análisis de impacto (conocer todos los componentes, administradores, usuarios, etc. impactados) ante un cambio o falla de un componente se debe marcar los nodos afectados y armar el árbol de alcance. El árbol incluirá todos los componentes impactados, los administradores y los usuarios.

En la figura 1 vemos ejemplos de cómo se modelan las relaciones entre componentes:

Ejemplo 1. Relación de dependencia simple. Vemos que "Usuario" depende de "Aplicación" y "Aplicación" depende de "Base de Datos".

Ejemplo 2. Balanceo de carga o alta disponibilidad de una aplicación. Vemos que "Aplicación Web" (1 y 2) dependen ambas de "Base de datos". Si llegara a fallar "Base de Datos" impactaría a ambas aplicaciones. En cambio "Usuario" depende de las dos "Aplicación Web" (1 y 2) en forma simultánea. En este caso si falla una de las "Aplicación Web", "Usuario" no se ve afectado. Es impactado si fallan ambas simultáneamente.

Ejemplo 3. Cluster Activo-Pasivo. Es una situación semejante a la anterior. Deben fallar simultáneamente "Base de datos 1" y "Base de Datos 2" para que Aplicación WEB sea afectada

Ejemplo 4. Cluster Activo-Pasivo. Aviso de caída de nodo activo. (CPN). En el ejemplo anterior, hemos visto que deben fallar ambos servidores de Base de datos para afectar a la aplicación WEB. Sin embargo, en ocasiones se requiere conocer el corte (pestañeo) que se produce entre la caída el elemento activo y la toma de control del elemento pasivo.

Para esto utilizaremos Coloured PetriNets [16], generando un token de "color" diferente para la caída del elemento activo. Al estudiar el podemos saber por el "color" del token si es un impacto completo o un simple "pestañeo" de la aplicación.

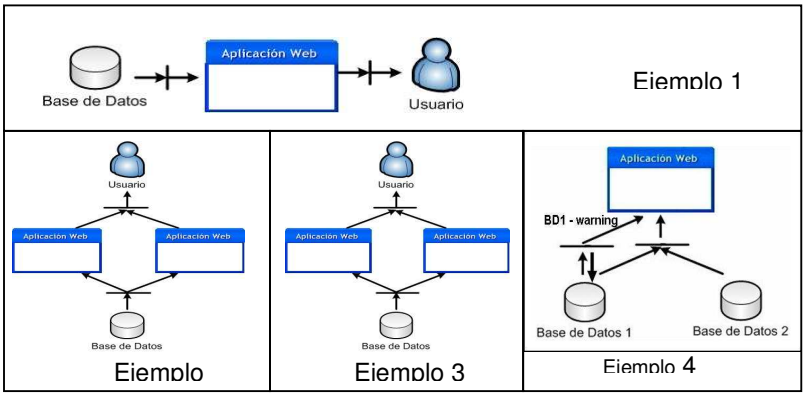

Fig. 1. Ejemplos de dependencias

Otras situaciones:

Tiempo. Hay ciertas dependencias que ocurren solamente en determinados momentos (Ej. Cierre de contabilidad, proceso que corre todos los 1º. de mes de 1 a.m. a 3 a. m.; o cierre de cajas, proceso diario que corre de lunes a viernes de 8 p.m. a 10 p.m.; etc).

Tenemos que resolver 2 cosas: especificar dependencias temporales (en un cierto intervalo de tiempo) y distinguir entre una dependencia común y una que es temporal.

Para especificar si el impacto es temporal, se pueden utilizar las Coloured PetriNets, en una forma similar a la planteada para el caso anterior. En este caso el Token especifica el intervalo en que existe la dependencia.

Problemas de red. Existen dificultades para representar las dependencias de red. El problema es que al impactarse un elemento de red se debe verificar que para todo par de componentes lógicos, que residen en equipos distintos y que existe una dependencia entre ellos, exista un camino de red (que no pase por los componentes de red afectados) entre los componentes de hardware donde reside cada componente lógico. En caso de no existir se debe incluir en el árbol de alcance el componente dependiente en la relación y continuar con el análisis de alcance. Por ejemplo, un "servidor WEB" que accede a una "Base de datos" residente en otro equipo. Si no hay comunicación entre ambos equipos, entonces "Servidor WEB" es afectado.

Para tratar este problema, se modela la red como otra PetriNet. Tenemos 2 PetriNets: la de dependencias y la que representa la red. El análisis de impacto debe realizarse en varias etapas. Primero analizar los impactos físicos (componentes de red afectados). Segundo, verificar que las dependencias de componentes lógicos en distintas máquinas tengan caminos (si no hay camino, se marcan los componentes lógicos dependientes) y por último continuar con el análisis de impacto.

Otra alternativa puede ser armar una única PetriNet, donde estén representados todos los componentes, incluyendo los de red. La diferencia es que se deben generar distintas combinaciones de transiciones que representan las fallas de distintos componentes de red. El problema es que esta PetriNet debe ser calculada ante cualquier cambio en la configuración de la red.

4. Caso de estudio

Hemos visto como podemos utilizar las PetriNets, para modelar la infraestructura de TI y realizar un análisis de impacto. Ahora veamos un ejemplo gráfico.

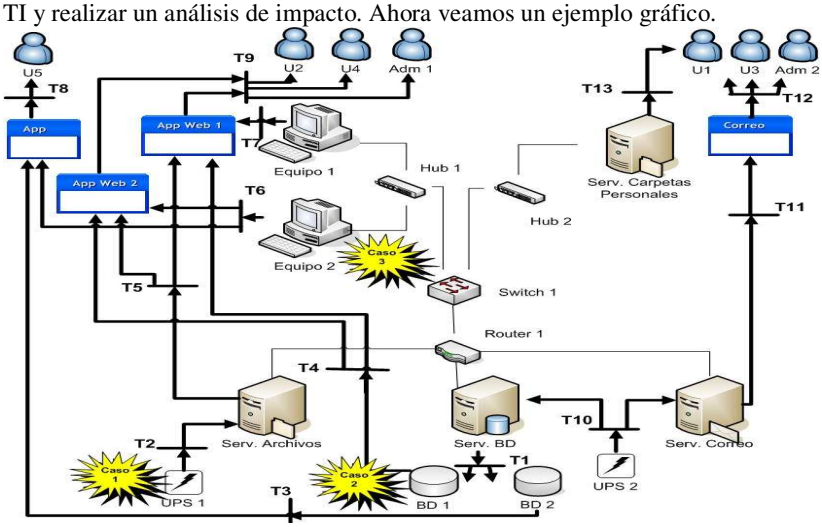

Fig. 2. Ejemplo de realidad

En la figura se pueden apreciar los servidores de BD, las base de datos BD1 y BD2, las aplicaciones Web AppWeb1 y AppWeb2 que utilizan BD1, y las relaciones de los distintos usuarios y administradores.

Hemos representado los "S-elementos" por los íconos que representan a cada tipo de nodo y hemos ya modelado las dependencias por medio de las transiciones.

4.2 Análisis de impacto

Analizaremos tres casos donde aplicaremos el algoritmo de impacto para ver cuáles son los componentes afectados.

4.2.1 Caso 1 – UPS 1

Caída de un componente eléctrico UPS 1. Se analizan los componentes afectados.

Como podemos apreciar en la fig. 3, los componentes impactados son UPS 1, Serv. Archivos, App Web1 y App Web2. Al fallar ambas aplicaciones se observa el impacto en U2, U4 y Adm1.

4.2.2 Caso 2 – BD 1

El segundo caso consiste en suponer la caída de un componente de software (BD 1) y analizar cuáles son los componentes involucrados (afectados).

En fig 3. vemos que los componentes impactados son BD 1, AppWeb 1, AppWeb 2, y al igual que el caso anterior al fallar ambas se impacta a U2, U4 y Adm1.

4.2.3 Caso 3 – Equipo 2

Este caso supone la falla de *Equipo 2* y analiza los componentes afectados.

Vemos que los componentes impactados son Equipo 2, App y AppWeb 2. Debido al impacto en App, se ve impactado U5, pero al fallar AppWeb 2 y no fallar simultáneamente AppWeb1 los usuarios U2, U3 y Adm1 no son impactados.

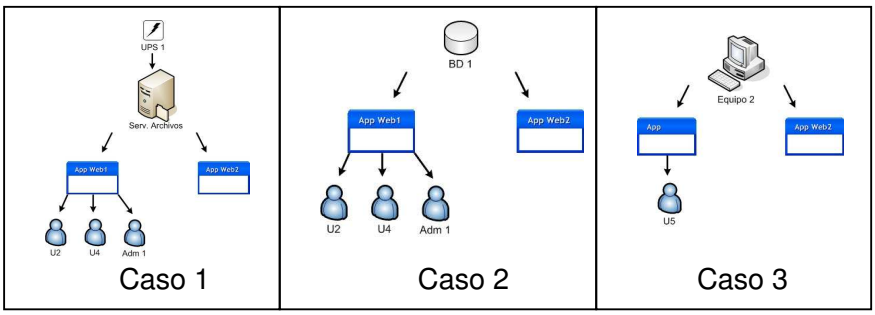

Fig. 3. Análisis de alcance de casos de estudio.

5. Conclusión y trabajos futuros

Hemos mostrado la necesidad de la gestión de configuraciones y de un análisis de impacto previo a implementar un cambio de configuración en la infraestructura tecnológica de una organización.

Hemos propuesto, representar la infraestructura con PetriNets incluyendo resolver diversas configuraciones complejas como clusters, alta disponibilidad, etc.

Aunque hay algunos problemas aún sin resolver, las Coloured PetriNets permitieron resolver diversos casos, como el del tiempo y notificaciones de "pestañeos" de un servicio (Ej. cuando cae el elemento activo de un cluster Activo-Pasivo).

El análisis de componentes de red se resuelve realizando un estudio en varias etapas.

Estos trabajos han sido la base para el desarrollo de un sistema de gestión de configuraciones y cambios de infraestructura que incluye una implementación de la CMDB según los lineamientos de ITIL. La herramienta permite registrar los componentes y sus relaciones (de red y dependencias) y realizar análisis de impacto ante cambios a alguno de los componentes de la infraestructura.

Nuestros planes incluyen profundizar en la especificación del tratamiento de tiempo y estudiar alternativas que permitan mejorar el tratamiento de la red.

Queremos utilizar esta representación de la CMDB para conocer los componentes necesarios para brindar un servicio, información básica para planificar los SLAs (Acuerdos de Niveles de Servicio). También queremos calcular la disponibilidad de un servicio, que en definitiva es afectado por la sumatoria de las indisponibilidades de cada uno de los componentes necesarios para brindarlo.

6. Referencias

1. COBIT. http://www.isaca.org/, Ult. acc. 2007.
2. ITIL. http://www.itsm.org/, http://www.itil.org, http://www.itilsurvival.com, Ult. acc. 2007.
3. S.Machuca, G.Sasco, N.Chiaro: Un modelo basado en grafos para análisis de impacto en cambios de componentes de infraestructura, JAIIO-ASIS 2005.
4. S.Machuca, G.Sasco, N.Chiaro: Análisis de impacto en la gestión de cambios de configuración de componentes de IT, CLAIO (Conferencia Latinoamericana de Investigación Operativa) 2006.
5. S.Machuca, G.Sasco, N.Chiaro: Análisis de impacto en la gestión de cambios de servicios de Telecomunicaciones, MVD TELCOM 2006 (I Congreso Regional de Telecomunicaciones).
6. S.Machuca, G.Sasco, N.Chiaro: H.Jiménez. Modelo de grafos para el estudio de la disponibilidad y la gestión de los Niveles de Servicio en servicios de IT, CACIC 2007 (XII Congreso Argentino de ciencias de la Computación).
7. Kunrong Chen, Václav Rajlich, Case Study of Feature Location Using Dependence Graph.. Department of Computer Science, Wayne State University, 2002, pp. 293 - 299.
8. Reisig W., Petri Nets, an introduction (EATCS Monographs on the Theoretical Computer Science Vol 4), Berlin Springer-Brlag, 1985.
9. Charles Thomas Betz, The convergence of metadata and IT service management, 2003.
10. M. Berkhout, R. Harrow, Service Support: Service Desk and the Process of Incident Management, Problem Management, Configuration Management, Change Management and Release Management, London, The Stationery Office, 2000.
11. A. Cassidy, K. Guggenberger A Practical Guide to Information Systems Process Improvement. Boca Raton, FL, St. Lucie, 2001.
12. IT Service Management Forum, IT Service Managment Forum, USA, Attendee FAQ, 2003.
13. J. Van Bon, G. Kemmerling, IT service management : an introduction, Canada, 2002.
14. MOF. http://www.microsoft.com/technet/itsolutions/cits/mo/mof/default.mspx, Ult. acc. 2006.
15. Erp4it: Managing information systems, http://erp4it.typepad.com/erp4it/2004/10/cmdb_chaos_and_.html, Ult. acc. 2006.
16. K. Jensen: A Brief Introduction to Coloured Petri Nets. In: E. Brinksma (ed.): Tools and Algorithms for the Construction and Analysis of Systems. Proceeding of the TACAS'97 Workshop, Enschede, The Netherlands 1997, Lecture Notes in Computer Science Vol. 1217, Springer-Verlag 1997, 203-208.